**עבודת גמר**

**בתכנון ותכנות מערכות**

**במסלול שירותי רשת אינטרנט**

**שנת הלימודים**

תשפ"א

|  |
| --- |
|  |
|  |
|  |
|  |

**שם התלמיד:** רז קיסוס

**בית ספר:** כדורי 280065

**ת.ז:** 213869753

**שם המורה:** עיזאת נגאר

# תוכן עניינים

[תוכן עניינים 2](#_Toc67409630)

[מבוא 3](#_Toc67409631)

[מבנה בסיס הנתונים של האתר 4](#_Toc67409632)

[מפת קשרי גומלין (DSD): 4](#_Toc67409633)

[רשימת הטבלאות ומבנה כל טבלה: 4](#_Toc67409634)

[מבנה התיקיות 5](#_Toc67409635)

[ממשקי המחלקות 6](#_Toc67409636)

[שירותי רשת 7](#_Toc67409637)

[שירותי רשת שהאפליקציה צורכת 7](#_Toc67409638)

[שירותי רשת שהאפליקציה מספקת 7](#_Toc67409639)

[מבנה בסיס הנתונים של שירות הרשת 7](#_Toc67409640)

[תמונת קשרי הגומלין מתוך מסד הנתונים של שירות הרשת (DSD) 7](#_Toc67409641)

[מבנה טבלאות שרות הרשת 7](#_Toc67409642)

[ממשקי מחלקות שירות הרשת 8](#_Toc67409643)

[מפת האתר 9](#_Toc67409644)

[צילום המסכים העיקריים באתר. 9](#_Toc67409645)

[מדריך האפליקציה למשתמש 10](#_Toc67409646)

[מדריך האפליקציה למנהל 10](#_Toc67409647)

[קוד האתר 11](#_Toc67409648)

[קוד שרות הרשת 12](#_Toc67409649)

[Read me 13](#_Toc67409650)

# מבוא

**מטרות האתר**

האתר משמש כגלריה אינטרנטית אשר תכיל תמונות של משתמש ותאפשר לו להעלות אותן ולשמור אותן תוך כדי שהוא יוכל לראות אותן באתר עצמו. האתר עצמו חוסך בהדפסת תמונות לאלבומים ובכך חוסך בנייר ובמקום אחסון.

**קהל יעד**

כלל קהל הצרכנים, כל אדם שאי פעם רצה לשמור תמונות באופן דיגיטלי ולראות אותן מכל מקום.

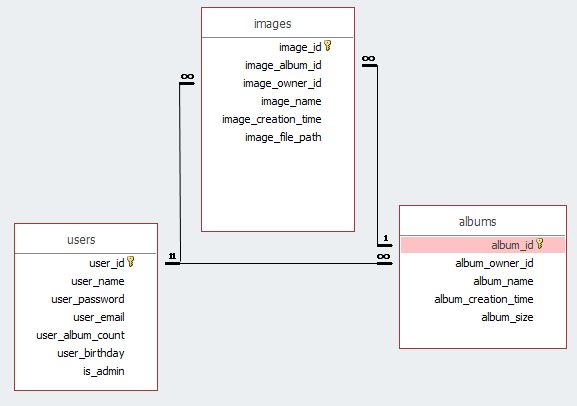
**תיאור האתר**

לא הבנתי...

# מבנה בסיס הנתונים של האתר

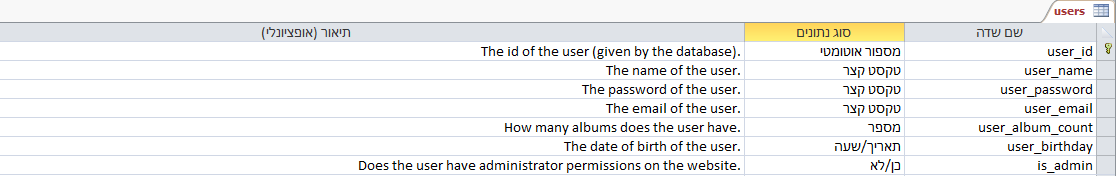
\* לא לבלבל עם מבנה הנתונים של שירות הרשת.

## מפת קשרי גומלין (DSD):

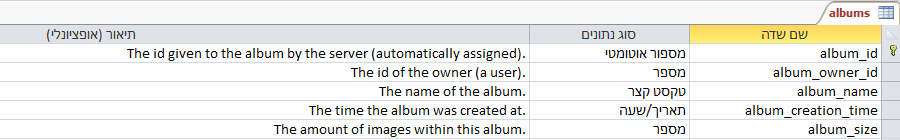


## רשימת הטבלאות ומבנה כל טבלה:

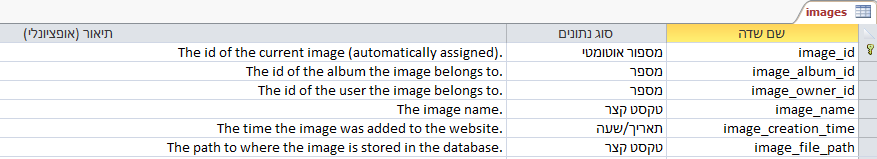
טבלת Users – מכילה את כל המשתמשים ומידע עליהם:



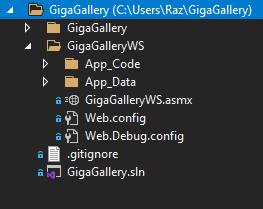
טבלת Albums – מכילה את כל האלבומים המאוכסנים בשרת ומידע עליהם:



טבלת Images – מכילה את כל התמונות המאוכסנות בשרת ומידע עליהן:



# מבנה התיקיות



**תיקיית GigaGallery:**

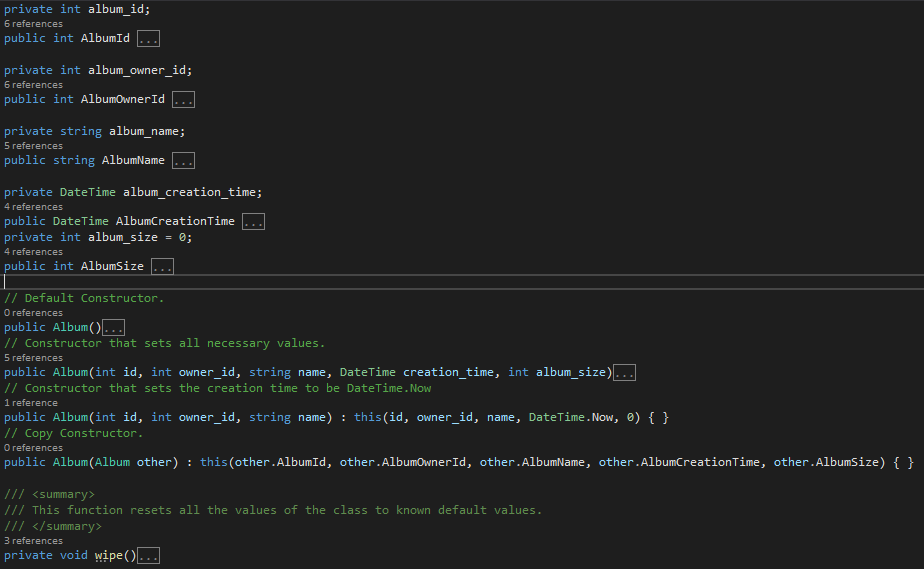
התיקייה מאחסנת את כל עמודי האתר (aspx) ואת הקוד שלהם, בנוסף כמה ספריות bootstrap ו-Javascript על מנת לסייע בעיצוב האתר.

**תיקיית GigaGalleryWS:**

תיקייה זו מאחסנת את כל הקשור לשירותי הרשת של האתר: Database, קוד צד שרת וכו...

# ממשקי המחלקות

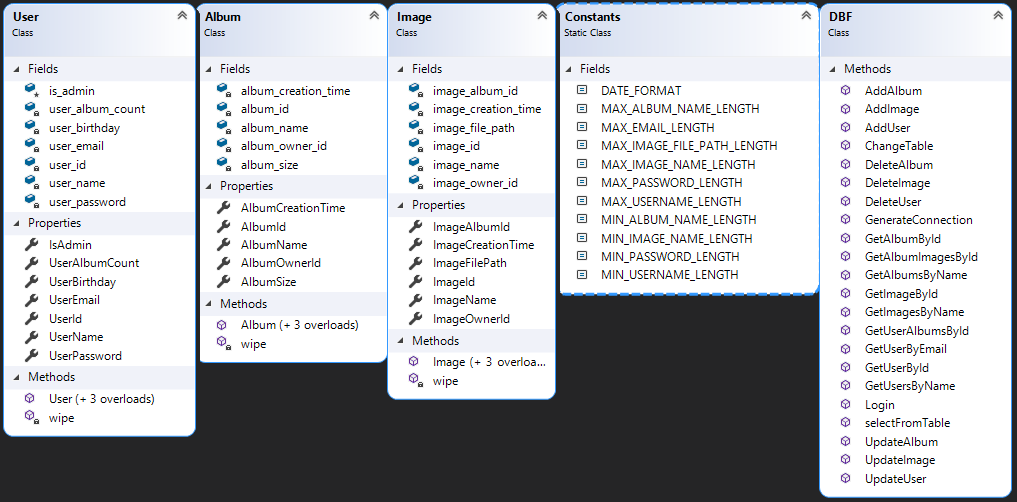
**מחלקת Album:**



# שירותי רשת

## הסבר על השרת.

# ממשקי מחלקות שירות הרשת

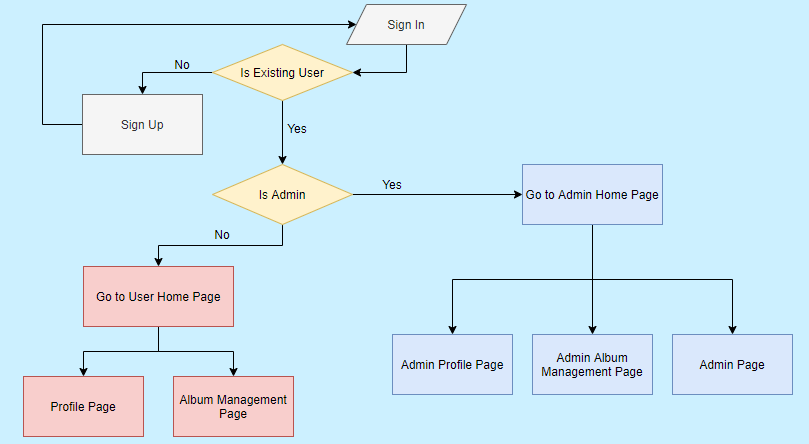


# מפת האתר

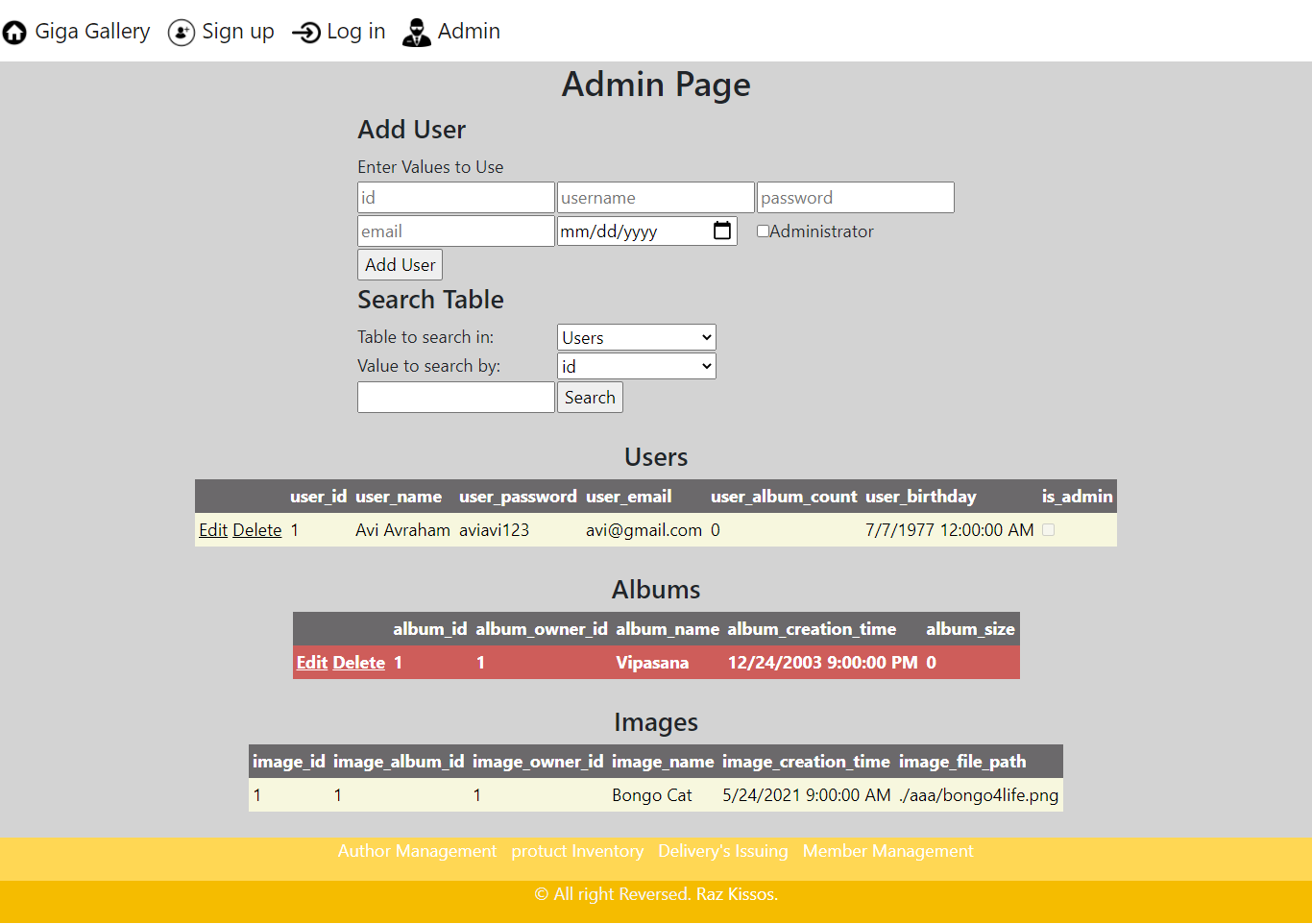
להלן מוצג תרשים הזרימה האפשרי באתר:

**שחור -** דפי מבקרים

**אדום -** דפי לקוחות

**כחול -** דפי מנהלים

## צילום המסכים העיקריים באתר.



# 

# קוד Admin Page:

<%@ Page Title="" Language="C#" MasterPageFile="~/AdminMasterPage.master" AutoEventWireup="true" CodeFile="AdminPage.aspx.cs" Inherits="AdminPage" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">

    <style type="text/css">

        .auto-style1 {

            width: 166px;

        }

        .grayDiv {

            background-color: #D3D3D3;

        }

        .auto-style2 {

            width: 177px;

        }

    </style>

    </asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceholder1" Runat="Server">

    <div class="grayDiv" align="center">

        <h2>Admin Page</h2>

        <table>

            <tr>

                <td class="auto-style2">

                    <h4>Add User</h4>

                </td>

            </tr>

            <tr>

                <td class="auto-style2">

                    <label>Enter Values to Use</label>

                </td>

            </tr>

            <tr>

                <td class="auto-style2">

                    <asp:TextBox ID="idTB" runat="server" placeholder="id" TextMode="Number"></asp:TextBox>

                </td>

                <td>

                    <asp:textbox ID="usernameTB" runat="server" placeholder="username"></asp:textbox>

                </td>

                <td>

                    <asp:TextBox ID="passwordTB" runat="server" placeholder="password"></asp:TextBox>

                </td>

            </tr>

            <tr>

                <td class="auto-style2">

                    <asp:TextBox ID="emailTB" runat="server" placeholder="email" TextMode="Email"></asp:TextBox>

                </td>

                <td>

                    <asp:TextBox ID="birthdayTB" runat="server" placeholder="birthday" TextMode="Date"></asp:TextBox>

                </td>

                <td>

                    <asp:CheckBox ID="isAdminCB" runat="server" Text="Administrator"/>

                </td>

            </tr>

            <tr>

                <td class="auto-style2">

                    <asp:Button runat="server" id="addUserBtn" onclick="addUserBtn\_Click" Text="Add User"></asp:Button>

                </td>

            </tr>

            <tr>

                <td class="auto-style2">

                    <h4>Search Table</h4>

                </td>

            </tr>

            <tr>

                <%// Find a way to get table names and column names from database and not hardcode it in asp. //%>

                <td class="auto-style2">

                    <label>Table to search in: </label>

                </td>

                <td>

                    <select id="TableSelector" class="auto-style1" >

                        <option>Users</option>

                        <option>Albums</option>

                        <option>Images</option>

                    </select>

                </td>

            <tr>

                <td>

                    <label>Value to search by: </label>

                </td>

                <td>

                    <select id="SerchBySelector" class="auto-style1" >

                        <option>id</option>

                        <option>username</option>

                        <option>date of birth</option>

                        <option>email</option>

                        <option>is admin</option>

                        <option>password</option>

                        <option>album count</option>

                    </select>

                </td>

            </tr>

            <tr>

                <td>

                    <input id="SearchBar">

                </td>

                <td>

                    <asp:Button runat="server" id="searchDataBtn" onclick="searchDataBtn\_Click" Text="Search"></asp:Button>

                </td>

            </tr>

        </table>

        <br />

        <h4 align="center">Users</h4>

        <asp:GridView ID="UsersGridView" runat="server" AutoGenerateColumns="False" BackColor="White" BorderColor="#DEDFDE" BorderStyle="None" BorderWidth="1px" CellPadding="4" DataKeyNames="user\_id" DataSourceID="Users" ForeColor="Black" GridLines="Vertical">

            <AlternatingRowStyle BackColor="White" />

            <Columns>

                <asp:CommandField ShowDeleteButton="True" ShowEditButton="True" />

                <asp:BoundField DataField="user\_id" HeaderText="user\_id" InsertVisible="False" ReadOnly="True" SortExpression="user\_id" />

                <asp:BoundField DataField="user\_name" HeaderText="user\_name" SortExpression="user\_name" />

                <asp:BoundField DataField="user\_password" HeaderText="user\_password" SortExpression="user\_password" />

                <asp:BoundField DataField="user\_email" HeaderText="user\_email" SortExpression="user\_email" />

                <asp:BoundField DataField="user\_album\_count" HeaderText="user\_album\_count" SortExpression="user\_album\_count" />

                <asp:BoundField DataField="user\_birthday" HeaderText="user\_birthday" SortExpression="user\_birthday" />

                <asp:CheckBoxField DataField="is\_admin" HeaderText="is\_admin" SortExpression="is\_admin" />

            </Columns>

            <FooterStyle BackColor="#CCCC99" />

            <HeaderStyle BackColor="#6B696B" Font-Bold="True" ForeColor="White" />

            <PagerStyle BackColor="#F7F7DE" ForeColor="Black" HorizontalAlign="Right" />

            <RowStyle BackColor="#F7F7DE" />

            <SelectedRowStyle BackColor="#CE5D5A" Font-Bold="True" ForeColor="White" />

            <SortedAscendingCellStyle BackColor="#FBFBF2" />

            <SortedAscendingHeaderStyle BackColor="#848384" />

            <SortedDescendingCellStyle BackColor="#EAEAD3" />

            <SortedDescendingHeaderStyle BackColor="#575357" />

        </asp:GridView>

        <asp:SqlDataSource ID="Users" runat="server" ConflictDetection="CompareAllValues" ConnectionString="<%$ ConnectionStrings:LaptopConnectionString %>" DeleteCommand="DELETE FROM [users] WHERE [user\_id] = ? AND [user\_name] = ? AND [user\_password] = ? AND [user\_email] = ? AND (([user\_album\_count] = ?) OR ([user\_album\_count] IS NULL AND ? IS NULL)) AND [user\_birthday] = ? AND [is\_admin] = ?" InsertCommand="INSERT INTO [users] ([user\_id], [user\_name], [user\_password], [user\_email], [user\_album\_count], [user\_birthday], [is\_admin]) VALUES (?, ?, ?, ?, ?, ?, ?)" OldValuesParameterFormatString="original\_{0}" ProviderName="<%$ ConnectionStrings:LaptopConnectionString.ProviderName %>" SelectCommand="SELECT \* FROM [users]" UpdateCommand="UPDATE [users] SET [user\_name] = ?, [user\_password] = ?, [user\_email] = ?, [user\_album\_count] = ?, [user\_birthday] = ?, [is\_admin] = ? WHERE [user\_id] = ? AND [user\_name] = ? AND [user\_password] = ? AND [user\_email] = ? AND (([user\_album\_count] = ?) OR ([user\_album\_count] IS NULL AND ? IS NULL)) AND [user\_birthday] = ? AND [is\_admin] = ?">

            <DeleteParameters>

                <asp:Parameter Name="original\_user\_id" Type="Int32" />

                <asp:Parameter Name="original\_user\_name" Type="String" />

                <asp:Parameter Name="original\_user\_password" Type="String" />

                <asp:Parameter Name="original\_user\_email" Type="String" />

                <asp:Parameter Name="original\_user\_album\_count" Type="Int32" />

                <asp:Parameter Name="original\_user\_album\_count" Type="Int32" />

                <asp:Parameter Name="original\_user\_birthday" Type="DateTime" />

                <asp:Parameter Name="original\_is\_admin" Type="Boolean" />

            </DeleteParameters>

            <InsertParameters>

                <asp:Parameter Name="user\_id" Type="Int32" />

                <asp:Parameter Name="user\_name" Type="String" />

                <asp:Parameter Name="user\_password" Type="String" />

                <asp:Parameter Name="user\_email" Type="String" />

                <asp:Parameter Name="user\_album\_count" Type="Int32" />

                <asp:Parameter Name="user\_birthday" Type="DateTime" />

                <asp:Parameter Name="is\_admin" Type="Boolean" />

            </InsertParameters>

            <UpdateParameters>

                <asp:Parameter Name="user\_name" Type="String" />

                <asp:Parameter Name="user\_password" Type="String" />

                <asp:Parameter Name="user\_email" Type="String" />

                <asp:Parameter Name="user\_album\_count" Type="Int32" />

                <asp:Parameter Name="user\_birthday" Type="DateTime" />

                <asp:Parameter Name="is\_admin" Type="Boolean" />

                <asp:Parameter Name="original\_user\_id" Type="Int32" />

                <asp:Parameter Name="original\_user\_name" Type="String" />

                <asp:Parameter Name="original\_user\_password" Type="String" />

                <asp:Parameter Name="original\_user\_email" Type="String" />

                <asp:Parameter Name="original\_user\_album\_count" Type="Int32" />

                <asp:Parameter Name="original\_user\_album\_count" Type="Int32" />

                <asp:Parameter Name="original\_user\_birthday" Type="DateTime" />

                <asp:Parameter Name="original\_is\_admin" Type="Boolean" />

            </UpdateParameters>

        </asp:SqlDataSource>

        <br />

        <h4 align="center">Albums</h4>

        <asp:GridView ID="AlbumGridView" runat="server" AutoGenerateColumns="False" BackColor="White" BorderColor="#DEDFDE" BorderStyle="None" BorderWidth="1px" CellPadding="4" DataKeyNames="album\_id" DataSourceID="Albums" ForeColor="Black" GridLines="Vertical">

            <AlternatingRowStyle BackColor="White" />

            <Columns>

                <asp:CommandField ShowDeleteButton="True" ShowEditButton="True" />

                <asp:BoundField DataField="album\_id" HeaderText="album\_id" InsertVisible="False" ReadOnly="True" SortExpression="album\_id" />

                <asp:BoundField DataField="album\_owner\_id" HeaderText="album\_owner\_id" SortExpression="album\_owner\_id" />

                <asp:BoundField DataField="album\_name" HeaderText="album\_name" SortExpression="album\_name" />

                <asp:BoundField DataField="album\_creation\_time" HeaderText="album\_creation\_time" SortExpression="album\_creation\_time" />

                <asp:BoundField DataField="album\_size" HeaderText="album\_size" SortExpression="album\_size" />

            </Columns>

            <FooterStyle BackColor="#CCCC99" />

            <HeaderStyle BackColor="#6B696B" Font-Bold="True" ForeColor="White" />

            <PagerStyle BackColor="#F7F7DE" ForeColor="Black" HorizontalAlign="Right" />

            <RowStyle BackColor="#F7F7DE" />

            <SelectedRowStyle BackColor="#CE5D5A" Font-Bold="True" ForeColor="White" />

            <SortedAscendingCellStyle BackColor="#FBFBF2" />

            <SortedAscendingHeaderStyle BackColor="#848384" />

            <SortedDescendingCellStyle BackColor="#EAEAD3" />

            <SortedDescendingHeaderStyle BackColor="#575357" />

        </asp:GridView>

         <asp:SqlDataSource ID="Albums" runat="server" ConflictDetection="CompareAllValues" ConnectionString="<%$ ConnectionStrings:LaptopConnectionString %>" DeleteCommand="DELETE FROM [albums] WHERE [album\_id] = ? AND [album\_owner\_id] = ? AND [album\_name] = ? AND [album\_creation\_time] = ? AND (([album\_size] = ?) OR ([album\_size] IS NULL AND ? IS NULL))" InsertCommand="INSERT INTO [albums] ([album\_id], [album\_owner\_id], [album\_name], [album\_creation\_time], [album\_size]) VALUES (?, ?, ?, ?, ?)" OldValuesParameterFormatString="original\_{0}" ProviderName="<%$ ConnectionStrings:LaptopConnectionString.ProviderName %>" SelectCommand="SELECT \* FROM [albums]" UpdateCommand="UPDATE [albums] SET [album\_owner\_id] = ?, [album\_name] = ?, [album\_creation\_time] = ?, [album\_size] = ? WHERE [album\_id] = ? AND [album\_owner\_id] = ? AND [album\_name] = ? AND [album\_creation\_time] = ? AND (([album\_size] = ?) OR ([album\_size] IS NULL AND ? IS NULL))">

            <DeleteParameters>

                <asp:Parameter Name="original\_album\_id" Type="Int32" />

                <asp:Parameter Name="original\_album\_owner\_id" Type="Int32" />

                <asp:Parameter Name="original\_album\_name" Type="String" />

                <asp:Parameter Name="original\_album\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="original\_album\_size" Type="Int32" />

                <asp:Parameter Name="original\_album\_size" Type="Int32" />

            </DeleteParameters>

            <InsertParameters>

                <asp:Parameter Name="album\_id" Type="Int32" />

                <asp:Parameter Name="album\_owner\_id" Type="Int32" />

                <asp:Parameter Name="album\_name" Type="String" />

                <asp:Parameter Name="album\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="album\_size" Type="Int32" />

            </InsertParameters>

            <UpdateParameters>

                <asp:Parameter Name="album\_owner\_id" Type="Int32" />

                <asp:Parameter Name="album\_name" Type="String" />

                <asp:Parameter Name="album\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="album\_size" Type="Int32" />

                <asp:Parameter Name="original\_album\_id" Type="Int32" />

                <asp:Parameter Name="original\_album\_owner\_id" Type="Int32" />

                <asp:Parameter Name="original\_album\_name" Type="String" />

                <asp:Parameter Name="original\_album\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="original\_album\_size" Type="Int32" />

                <asp:Parameter Name="original\_album\_size" Type="Int32" />

            </UpdateParameters>

        </asp:SqlDataSource>

        <br />

        <h4 align="center">Images</h4>

        <asp:GridView ID="ImagesGridView" runat="server" AutoGenerateColumns="False" BackColor="White" BorderColor="#DEDFDE" BorderStyle="None" BorderWidth="1px" CellPadding="4" DataKeyNames="image\_id" DataSourceID="Images" ForeColor="Black" GridLines="Vertical">

            <AlternatingRowStyle BackColor="White" />

            <Columns>

                <asp:CommandField ShowDeleteButton="True" ShowEditButton="True" />

                <asp:BoundField DataField="image\_id" HeaderText="image\_id" InsertVisible="False" ReadOnly="True" SortExpression="image\_id" />

                <asp:BoundField DataField="image\_album\_id" HeaderText="image\_album\_id" SortExpression="image\_album\_id" />

                <asp:BoundField DataField="image\_owner\_id" HeaderText="image\_owner\_id" SortExpression="image\_owner\_id" />

                <asp:BoundField DataField="image\_name" HeaderText="image\_name" SortExpression="image\_name" />

                <asp:BoundField DataField="image\_creation\_time" HeaderText="image\_creation\_time" SortExpression="image\_creation\_time" />

                <asp:BoundField DataField="image\_file\_path" HeaderText="image\_file\_path" SortExpression="image\_file\_path" />

            </Columns>

            <FooterStyle BackColor="#CCCC99" />

            <HeaderStyle BackColor="#6B696B" Font-Bold="True" ForeColor="White" />

            <PagerStyle BackColor="#F7F7DE" ForeColor="Black" HorizontalAlign="Right" />

            <RowStyle BackColor="#F7F7DE" />

            <SelectedRowStyle BackColor="#CE5D5A" Font-Bold="True" ForeColor="White" />

            <SortedAscendingCellStyle BackColor="#FBFBF2" />

            <SortedAscendingHeaderStyle BackColor="#848384" />

            <SortedDescendingCellStyle BackColor="#EAEAD3" />

            <SortedDescendingHeaderStyle BackColor="#575357" />

        </asp:GridView>

        <asp:SqlDataSource ID="Images" runat="server" ConnectionString="<%$ ConnectionStrings:LaptopConnectionString %>" ProviderName="<%$ ConnectionStrings:LaptopConnectionString.ProviderName %>" SelectCommand="SELECT \* FROM [images]" ConflictDetection="CompareAllValues" DeleteCommand="DELETE FROM [images] WHERE [image\_id] = ? AND (([image\_album\_id] = ?) OR ([image\_album\_id] IS NULL AND ? IS NULL)) AND (([image\_owner\_id] = ?) OR ([image\_owner\_id] IS NULL AND ? IS NULL)) AND (([image\_name] = ?) OR ([image\_name] IS NULL AND ? IS NULL)) AND (([image\_creation\_time] = ?) OR ([image\_creation\_time] IS NULL AND ? IS NULL)) AND [image\_file\_path] = ?" InsertCommand="INSERT INTO [images] ([image\_id], [image\_album\_id], [image\_owner\_id], [image\_name], [image\_creation\_time], [image\_file\_path]) VALUES (?, ?, ?, ?, ?, ?)" OldValuesParameterFormatString="original\_{0}" UpdateCommand="UPDATE [images] SET [image\_album\_id] = ?, [image\_owner\_id] = ?, [image\_name] = ?, [image\_creation\_time] = ?, [image\_file\_path] = ? WHERE [image\_id] = ? AND (([image\_album\_id] = ?) OR ([image\_album\_id] IS NULL AND ? IS NULL)) AND (([image\_owner\_id] = ?) OR ([image\_owner\_id] IS NULL AND ? IS NULL)) AND (([image\_name] = ?) OR ([image\_name] IS NULL AND ? IS NULL)) AND (([image\_creation\_time] = ?) OR ([image\_creation\_time] IS NULL AND ? IS NULL)) AND [image\_file\_path] = ?">

            <DeleteParameters>

                <asp:Parameter Name="original\_image\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_album\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_album\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_owner\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_owner\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_name" Type="String" />

                <asp:Parameter Name="original\_image\_name" Type="String" />

                <asp:Parameter Name="original\_image\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="original\_image\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="original\_image\_file\_path" Type="String" />

            </DeleteParameters>

            <InsertParameters>

                <asp:Parameter Name="image\_id" Type="Int32" />

                <asp:Parameter Name="image\_album\_id" Type="Int32" />

                <asp:Parameter Name="image\_owner\_id" Type="Int32" />

                <asp:Parameter Name="image\_name" Type="String" />

                <asp:Parameter Name="image\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="image\_file\_path" Type="String" />

            </InsertParameters>

            <UpdateParameters>

                <asp:Parameter Name="image\_album\_id" Type="Int32" />

                <asp:Parameter Name="image\_owner\_id" Type="Int32" />

                <asp:Parameter Name="image\_name" Type="String" />

                <asp:Parameter Name="image\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="image\_file\_path" Type="String" />

                <asp:Parameter Name="original\_image\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_album\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_album\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_owner\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_owner\_id" Type="Int32" />

                <asp:Parameter Name="original\_image\_name" Type="String" />

                <asp:Parameter Name="original\_image\_name" Type="String" />

                <asp:Parameter Name="original\_image\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="original\_image\_creation\_time" Type="DateTime" />

                <asp:Parameter Name="original\_image\_file\_path" Type="String" />

            </UpdateParameters>

        </asp:SqlDataSource>

        <br />

    </div>

</asp:Content>

# מדריך האפליקציה למשתמש

משתמש שאינו מחובר יגיע לדף הבית וממנו יראה אפשרויות לגשת להתחברות, להירשמות, לגלריה שלו ולפרופיל שלו, כיוון שהוא אינו מחובר כל לחיצה על עמוד שדורש התחברות תעביר את המשתמש לדף הכניסה וממנו יוכל לגשת גם לעמוד ההירשמות. לאחר שהמשתמש התחבר בהצלחה הוא יוכל לבקר בכל העמודים המוצגים עבורו. בעמוד הפרופיל יוכל לשנות את השם והסיסמה שלו, ובעמוד הגלריה יוכל לערוך את האלבומים שלו.

# מדריך האפליקציה למנהל

עבור המנהל פתוחים בדיוק אותם העמודים שפתוחים למשתמשים רגילים אך כעת ייתווסף עבורו עמוד חדש הקרוי Admin Page אשר דרכו יכול המנהל לערוך את המידע של כל משתמש, ליצור משתמשים חדשים ולערוך מידע בטבלאות השונות ב-Database.

# קוד האתר

DBF:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Data;

using System.Data.OleDb;

using ConstantsNS;

using UserNS;

using AlbumNS;

using ImageNS;

/// <summary>

/// Summary description for DBF

/// </summary>

namespace DBFNS

{

    public class DBF

    {

        #region Update

        public static bool UpdateUser(User u)

        {

            if (GetUserById(u.UserId) != null)

            {

                string sql = string.Format("update users user\_name = '{0}' user\_password = '{1}' user\_email = '{2}' user\_album\_count = '{3}' user\_birthday = #{4}# is\_admin = '{5}' where user\_id = '{6}'", u.UserName, u.UserPassword, u.UserEmail, u.UserAlbumCount, u.UserAlbumCount, u.UserBirthday, u.IsAdmin, u.UserId);

                return ChangeTable(sql) == 1;

            }

            return false;

        }

        public static bool UpdateAlbum(Album al)

        {

            if (GetAlbumById(al.AlbumId) != null)

            {

                string sql = string.Format("update albums album\_name = '{0}' album\_owner\_id = '{1}' album\_creation\_time = #{2}# album\_size = '{3}' where album\_id = '{4}'", al.AlbumName, al.AlbumOwnerId, al.AlbumCreationTime, al.AlbumSize, al.AlbumId);

                return ChangeTable(sql) == 1;

            }

            return false;

        }

        public static bool UpdateImage(Image img)

        {

            if (GetImageById(img.ImageId) != null)

            {

                string sql = string.Format("update images image\_album\_id = '{0}' image\_owner\_id = '{1}' image\_name = '{2}' image\_creation\_time = #{3}# image\_file\_path = '{4}' where image\_id = '{5}'", img.ImageAlbumId, img.ImageOwnerId, img.ImageName, img.ImageCreationTime, img.ImageFilePath, img.ImageId);

                return ChangeTable(sql) == 1;

            }

            return false;

        }

        #endregion

        #region Delete

        public static bool DeleteUser(User u)

        {

            if (GetUserById(u.UserId) != null)

            {

                string sql = string.Format("delete from users where user\_id = '{0}'", u.UserId);

                return ChangeTable(sql) == 1;

            }

            return false;

        }

        public static bool DeleteAlbum(Album al)

        {

            if (GetAlbumById(al.AlbumId) != null)

            {

                string sql = string.Format("delete from albums where album\_id = '{0}'", al.AlbumId);

                return ChangeTable(sql) == 1;

            }

            return false;

        }

        public static bool DeleteImage(Image img)

        {

            if (GetImageById(img.ImageId) != null)

            {

                string sql = string.Format("delete from images where image\_id = '{0}'", img.ImageId);

                return ChangeTable(sql) == 1;

            }

            return false;

        }

        #endregion

        #region Select

        public static User GetUserById(int user\_id)

        {

            string sql = string.Format("select \* from users where user\_id='{0}' ", user\_id);

            DataRow dr;

            try

            {

                dr = selectFromTable(sql).Rows[0];

                return new User(int.Parse(dr["user\_id"].ToString()), dr["user\_name"].ToString(), dr["user\_email"].ToString(), dr["user\_password"].ToString(), DateTime.Parse(dr["user\_birthday"].ToString()), int.Parse(dr["user\_album\_count"].ToString()), bool.Parse(dr["is\_admin"].ToString()));

            }

            catch { return null; }

        }

        public static User GetUserByEmail(string email)

        {

            string sql = string.Format("select \* from users where user\_email='{0}' ", email);

            DataRow dr;

            try

            {

                dr = selectFromTable(sql).Rows[0];

                return new User(int.Parse(dr["user\_id"].ToString()), dr["user\_name"].ToString(), dr["user\_email"].ToString(), dr["user\_password"].ToString(), DateTime.Parse(dr["user\_birthday"].ToString()), int.Parse(dr["user\_album\_count"].ToString()), bool.Parse(dr["is\_admin"].ToString()));

            }

            catch { return null; }

        }

        public static Album GetAlbumById(int album\_id)

        {

            string sql = string.Format("select \* from albums where album\_id='{0}' ", album\_id);

            DataRow dr;

            try

            {

                dr = selectFromTable(sql).Rows[0];

                return new Album(int.Parse(dr["album\_id"].ToString()), int.Parse(dr["album\_owner\_id"].ToString()), dr["album\_name"].ToString(), DateTime.Parse(dr["album\_creation\_time"].ToString()), int.Parse(dr["album\_size"].ToString()));

            }

            catch { return null; }

        }

        public static Image GetImageById(int image\_id)

        {

            string sql = string.Format("select \* from images where image\_id='{0}' ", image\_id);

            DataRow dr;

            try

            {

                dr = selectFromTable(sql).Rows[0];

                return new Image(int.Parse(dr["image\_id"].ToString()), int.Parse(dr["owner\_id"].ToString()), int.Parse(dr["album\_id"].ToString()), dr["image\_name"].ToString(), DateTime.Parse(dr["image\_creation\_time"].ToString()), dr["image\_file"].ToString());

            }

            catch { return null; }

        }

        public static List<Album> GetUserAlbumsById(int owner\_id)

        {

            string sql = string.Format("select \* from albums where album\_owner\_id='{0}' ", owner\_id);

            DataTable dt;

            try

            {

                dt = selectFromTable(sql);

            }

            catch

            {

                return null;

            }

            if (dt.Rows.Count == 0)

                return null;

            List<Album> toRet = new List<Album>();

            for (int i = 0; i < dt.Rows.Count; i++)

            {

                DataRow dr = dt.Rows[i];

                toRet.Add(new Album(int.Parse(dr["album\_id"].ToString()), int.Parse(dr["album\_owner\_id"].ToString()), dr["album\_name"].ToString(), DateTime.Parse(dr["album\_creation\_time"].ToString()), int.Parse(dr["album\_size"].ToString())));

            }

            return toRet;

        }

        public static List<Image> GetAlbumImagesById(int album\_id)

        {

            string sql = string.Format("select \* from images where album\_id='{0}' ", album\_id);

            DataTable dt;

            try

            {

                dt = selectFromTable(sql);

            }

            catch

            {

                return null;

            }

            if (dt.Rows.Count == 0)

                return null;

            List<Image> toRet = new List<Image>();

            for (int i = 0; i < dt.Rows.Count; i++)

            {

                DataRow dr = dt.Rows[i];

                toRet.Add(new Image(int.Parse(dr["image\_id"].ToString()), int.Parse(dr["owner\_id"].ToString()), int.Parse(dr["album\_id"].ToString()), dr["image\_name"].ToString(), DateTime.Parse(dr["image\_creation\_time"].ToString()), dr["image\_file"].ToString()));

            }

            return toRet;

        }

        public static List<User> GetUsersByName(string user\_name)

        {

            string sql = string.Format("select \* from users where user\_name='{0}' ", user\_name);

            DataTable dt = selectFromTable(sql);

            List<User> listToRet = new List<User>();

            if (dt.Rows.Count == 0)

                return null;

            foreach (DataRow dr in dt.Rows)

            {

                listToRet.Add(new User(int.Parse(dr["user\_id"].ToString()), dr["user\_name"].ToString(), dr["user\_email"].ToString(), dr["user\_password"].ToString(), DateTime.Parse(dr["user\_birthday"].ToString()), int.Parse(dr["user\_album\_count"].ToString()), bool.Parse(dr["is\_admin"].ToString())));

            }

            return listToRet;

        }

        public static List<Album> GetAlbumsByName(string album\_name)

        {

            string sql = string.Format("select \* from albums where album\_name='{0}' ", album\_name);

            DataTable dt = selectFromTable(sql);

            List<Album> listToRet = new List<Album>();

            if (dt.Rows.Count == 0)

                return null;

            foreach (DataRow dr in dt.Rows)

            {

                listToRet.Add(new Album(int.Parse(dr["album\_id"].ToString()), int.Parse(dr["album\_owner\_id"].ToString()), dr["album\_name"].ToString(), DateTime.Parse(dr["album\_creation\_time"].ToString()), int.Parse(dr["album\_size"].ToString())));

            }

            return listToRet;

        }

        public static List<Image> GetImagesByName(string image\_name)

        {

            string sql = string.Format("select \* from images where image\_name='{0}' ", image\_name);

            DataTable dt = selectFromTable(sql);

            List<Image> listToRet = new List<Image>();

            if (dt.Rows.Count == 0)

                return null;

            foreach (DataRow dr in dt.Rows)

            {

                listToRet.Add(new Image(int.Parse(dr["image\_id"].ToString()), int.Parse(dr["image\_a;bum\_id"].ToString()), int.Parse(dr["image\_owner\_id"].ToString()), dr["image\_name"].ToString(), DateTime.Parse(dr["image\_creation\_time"].ToString()), dr["image\_file\_path"].ToString()));

            }

            return listToRet;

        }

        public static bool Login(string email, string password)

        {

            User u = GetUserByEmail(email);

            if (u != null)

            {

                if (password == u.UserPassword)

                    return true;

            }

            return false;

        }

        public static DataTable GetUsersTable()

        {

            string sql = string.Format("select \* from [users]");

            DataTable dt = new DataTable();

            dt = DBF.selectFromTable(sql);

            return dt;

        }

        #endregion

        #region Add

        public static bool AddUser(User u)

        {

            User users = GetUserByEmail(u.UserEmail);

            if (users != null)

            {

                throw new Exception("Email is already in use!");

            }

            string sql = string.Format("insert into users (user\_name, user\_password, user\_email, user\_birthday, user\_album\_count, is\_admin) values('{0}', '{1}', '{2}', #{3}#, '{4}' ,'{5}')", u.UserName, u.UserPassword, u.UserEmail, u.UserBirthday, u.UserAlbumCount, u.IsAdmin);

            return DBF.ChangeTable(sql) == 1;

        }

        public static bool AddAlbum(Album a)

        {

            // Get all albums with the same name and check if their owner is identical.

            List<Album> albums\_to\_check = GetAlbumsByName(a.AlbumName);

            if (albums\_to\_check != null)

            {

                foreach (Album al in albums\_to\_check)

                {

                    if (al.AlbumOwnerId == a.AlbumOwnerId)

                    {

                        // Code will stop if it reaces this becuase we throw exception.

                        throw new Exception("An Album with the same name already belongs to this user!");

                    }

                }

            }

            string sql = string.Format("insert into albums (album\_owner\_id, album\_name, album\_creation\_time, album\_size) values('{0}', '{1}', #{2}#, '{3}')", a.AlbumOwnerId, a.AlbumName, a.AlbumCreationTime, a.AlbumSize);

            return DBF.ChangeTable(sql) == 1;

        }

        public static bool AddImage(Image i)

        {

            // Get all images with the same name and check if their album is identical.

            List<Image> images\_to\_check = GetImagesByName(i.ImageName);

            if (images\_to\_check != null)

            {

                foreach (Image img in images\_to\_check)

                {

                    if (img.ImageAlbumId == i.ImageAlbumId)

                    {

                        // Code will stop if it reaces this becuase we throw exception.

                        throw new Exception("An Image with the same name already exists in this album!");

                    }

                }

            }

            string sql = string.Format("insert into images (image\_album\_id, image\_owner\_id, image\_name, image\_creation\_time, image\_file\_path) values('{0}', '{1}', '{2}', #{3}#, '{4}')", i.ImageAlbumId, i.ImageOwnerId, i.ImageName, i.ImageCreationTime, i.ImageFilePath);

            return DBF.ChangeTable(sql) == 1;

        }

        #endregion

        #region general operations

        public static string GetConnectionString()

        {

            return @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + HttpContext.Current.Server.MapPath("App\_Data/GigaGallery.accdb");

        }

        public static OleDbConnection GenerateConnection()

        {

            // פעולה מקבלת שם קובץ של מסד נתונים ובונה אובייקט התחברות ופותחת ערוץ תקשורת

            OleDbConnection obj = new OleDbConnection();

            // Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Users\user\GigaGallery\GigaGalleryWS\App\_Data\GigaGallery.accdb

            obj.ConnectionString = GetConnectionString();

            obj.Open();

            return obj;

        }

        public static int ChangeTable(string sqlstring)

        {

            OleDbConnection conobj = GenerateConnection();

            OleDbCommand com = new OleDbCommand(sqlstring, conobj);

            int result = 0;

            try

            {

                result = com.ExecuteNonQuery();

            }

            catch (Exception e)

            {

                Console.WriteLine(e);

            }

            finally

            {

                conobj.Close();

            }

            return result;

        }

        public static DataTable selectFromTable(string sqlstring)

        {

            OleDbConnection obj = GenerateConnection();

            OleDbDataAdapter da = new OleDbDataAdapter(sqlstring, obj);

            DataTable dt = new DataTable();

            try

            {

                da.Fill(dt);

            }

            catch (Exception e)

            {

                Console.WriteLine(e);

            }

            finally

            {

                obj.Close();

            }

            return dt;

        }

        public static DataTable GetDBSchema()

        {

            OleDbConnection conobj = GenerateConnection();

            DataTable schema = conobj.GetSchema("Columns");

            conobj.Close();

            return schema;

        }

        #endregion

    }

}

Admin page:

 protected void addUserBtn\_Click(object sender, EventArgs e)

{

    this.errorLabel.Visible = false;

    string username = this.usernameTB.Text;

    string email = this.emailTB.Text;

    string password = this.passwordTB.Text;

    bool isAdmin = this.isAdminCB.Checked;

    DateTime birthday = default(DateTime);

    try

    {

        birthday = DateTime.Parse(this.birthdayTB.Text);

    }

    catch

    {

        this.errorLabel.Text = "Error!";

        this.errorLabel.Visible = true;

        return;

    }

    // Validate Data:

    localhost.UserValidationWS validationWS = new localhost.UserValidationWS();

    if (!validationWS.isUsernameValid(username))

    {

        this.errorLabel.Text = validationWS.usernameLengthInvalidMessage();

        this.errorLabel.Visible = true;

        return;

    }

    if (!validationWS.isEmailLengthValid(email))

    {

        this.errorLabel.Text = validationWS.emailLengthInvalidMessage();

        this.errorLabel.Visible = true;

        return;

    }

    if (!validationWS.isPasswordValid(password))

    {

        this.errorLabel.Text = validationWS.passwordLengthInvalidMessage();

        this.errorLabel.Visible = true;

        return;

    }

    localhost.GigaGalleryWS dbWS = new localhost.GigaGalleryWS();

    try

    {

        dbWS.AdminAddUser(username, email, password, birthday, isAdmin);

    }

    catch

    {

        this.errorLabel.Text = "Error!";

        this.errorLabel.Visible = true;

    }

    Response.Redirect(Page.Request.Url.ToString(), true);

}

# קוד שרות הרשת

Giga Gallery WS:

public bool CheckPasswordLength(string password)

{

    return password.Length >= Constants.MIN\_PASSWORD\_LENGTH && password.Length <= Constants.MAX\_PASSWORD\_LENGTH;

}

public bool CheckEmailLength(string email)

{

    return email.Length <= Constants.MAX\_EMAIL\_LENGTH;

}

[WebMethod]

public bool Login(string email, string password)

{

    // This function should only be called after we verified the password and email length!

    if (!CheckPasswordLength(password))

    {

        Exception e = new Exception("Password Error");

        e.Data.Add("stringInfo", string.Format("Password Length must be longer than {0} and shorter than {1}!", Constants.MIN\_PASSWORD\_LENGTH, Constants.MAX\_PASSWORD\_LENGTH));

        throw e;

    }

    else if (!CheckEmailLength(email))

    {

        Exception e = new Exception("Email Error");

        e.Data.Add("stringInfo", string.Format("Email Length must be shorter than {0}!", Constants.MAX\_EMAIL\_LENGTH));

        throw e;

    }

    return DBF.Login(email, password);

}

//[WebMethod]

//public Dictionary<string, string[]> GetDBSchema()

//{

//    DataTable schema = DBF.GetDBSchema();

//    foreach (DataRow row in schema.Rows)

//        Console.WriteLine("TABLE:" + row.Field<string>("TABLE\_NAME") + " COLUMN:" + row.Field<string>("COLUMN\_NAME"));

//        //TODO: separate the DataTable into dictionary and return int.

//    return null;

//}

[WebMethod]

public string GetConnectionString()

{

    return @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + HttpContext.Current.Server.MapPath("App\_Data/GigaGallery.accdb");

}

[WebMethod]

public bool Signup(string name, string email, string password, DateTime birthday)

{

    try

    {

        // `id` doesn't matter in the DBF function AddUser because database gives id automatically.

        User u = new User(1, name, email, password, birthday);

        bool dpfRes = DBF.AddUser(u);

    }

    catch

    {

        return false;

    }

    return true;

}

[WebMethod]

public User GetUserObj(string email)

{

    return DBF.GetUserByEmail(email);

}

[WebMethod]

public bool AdminAddUser(string name, string email, string password, DateTime birthday, bool isadmin)

{

    try

    {

        // `id` doesn't matter in the DBF function AddUser because database gives id automatically.

        User u = new User(1, name, email, password, birthday, 0, isadmin);

        return DBF.AddUser(u);

    }

    catch (Exception e){ throw e; }

}

[WebMethod]

public bool AddAlbum(int owner\_id, string name)

{

    try

    {

        // Album id doesn't matter since it is not considered in the AddAlbum function in DBF.

        Album al = new Album(1, owner\_id, name);

        return DBF.AddAlbum(al);

    }

    catch(Exception e) { throw e; }

}

[WebMethod]

public bool AddImage(int owner\_id, int album\_id, string name, byte[] bytes)

{

    try

    {

        // TODO: Compute file path and save bytes as image.

        // Image id doesn't matter since it is not considered in the AddImage function in DBF.

        //string file\_path = ""

        //Image img = new Image(1, owner\_id, album\_id, name);

        //return DBF.AddAlbum(al);

        return false;

    }

    catch (Exception e) { throw e; }

}

[WebMethod ]

public DataTable GetAllUsers()

{

    DataTable dt = new DataTable();

    dt = DBF.GetUsersTable();

    dt.TableName = "usersDataTable";

    return dt;

}

User Validation WS:

[WebMethod]

public bool isPasswordValid(string password)

{

    return password.Length >= Constants.MIN\_PASSWORD\_LENGTH && password.Length <= Constants.MAX\_PASSWORD\_LENGTH;

}

[WebMethod]

public string passwordLengthInvalidMessage()

{

    return string.Format("Password Length Invalid! needs to be between {0} and {1}!", Constants.MIN\_PASSWORD\_LENGTH, Constants.MAX\_PASSWORD\_LENGTH);

}

[WebMethod]

public bool isEmailLengthValid(string email)

{

    return email.Length <= Constants.MAX\_EMAIL\_LENGTH;

}

[WebMethod]

public string emailLengthInvalidMessage()

{

    return string.Format("Email Length invalid! needs to be equal to or lower than {0}!", Constants.MAX\_EMAIL\_LENGTH);

}

[WebMethod]

public bool isUsernameValid(string username)

{

    return username.Length >= Constants.MIN\_USERNAME\_LENGTH && username.Length <= Constants.MAX\_USERNAME\_LENGTH;

}

[WebMethod]

public string usernameLengthInvalidMessage()

{

    return string.Format("Username Length invalid! needs to be between {0} and {1}!", Constants.MIN\_USERNAME\_LENGTH, Constants.MAX\_USERNAME\_LENGTH);

}

[WebMethod]

public bool isAlbumNameValid(string albumName)

{

    return albumName.Length >= Constants.MIN\_ALBUM\_NAME\_LENGTH && albumName.Length <= Constants.MAX\_ALBUM\_NAME\_LENGTH;

}

[WebMethod]

public bool isImageNameValid(string imageName)

{

    return imageName.Length >= Constants.MIN\_IMAGE\_NAME\_LENGTH && imageName.Length <= Constants.MAX\_IMAGE\_NAME\_LENGTH;

}

# Read me

לצורך כניסה בתור משתמש (רגיל) רשום באתר יש למלא את הפרטים הבאים בLogin :

**שם משתמש:** username

**סיסמא:** password

לצורך כניסה בתור מנהל האתר יש למלא את הפרטים הבאים בLogin :

**שם משתמש**: admin

**סיסמא:** password