Cloud of Sound

G:\Project\Images\Cloud.png

**עבודת גמר  
בתכנון ותכנות מערכות  
המסלול שירותי רשת אינטרנט  
שנת הלימודים תשע"ה**

שם התלמיד: דן אלעזרי  
בית ספר: נבון  
ת.ז: 209044973  
שם המורים: מירי באורק ומנחם גנץ

תוכן עניינים

[הקדמה: 3](#_Toc422782340)

[תאור קצר של האתר ומטרותיו: 3](#_Toc422782341)

[תאור קצר של המערכת: 3](#_Toc422782342)

[קהל יעד 3](#_Toc422782343)

[ניתוח המערכת – UML 4](#_Toc422782344)

[מבנה בסיס הנתונים 5](#_Toc422782345)

[קשרי גומלין (DSD) 6](#_Toc422782346)

[מבנה התיקיות 6](#_Toc422782347)

[מבנה התיקיות – הסבר 6](#_Toc422782348)

[המחלקות 7](#_Toc422782349)

[שרותי הרשת באתר 7](#_Toc422782350)

[מבנה בסיס הנתונים של שירות הרשת 7](#_Toc422782351)

[**קשרי גומלין** 7](#_Toc422782352)

[מפת האתר 8](#_Toc422782353)

[מדריך האפליקציה 9](#_Toc422782354)

[אורח 9](#_Toc422782355)

[משתמש רשום 13](#_Toc422782356)

[אדמין 14](#_Toc422782357)

[דפי קוד 16](#_Toc422782358)

[מחלקות 16](#_Toc422782359)

[WebService 26](#_Toc422782360)

[דפים 28](#_Toc422782361)

[ReadMe 45](#_Toc422782362)

# **הקדמה:**

**נושא הפרוייקט:**

Cloud of sound – העלאה ושיתוף של קבצי אודיו.

## **תאור קצר של האתר ומטרותיו:**

אתר המאפשר העלאת קבצי אודיו, הקשבה להם ושיתופם.

האתר גם מאפשר שמירה של קבצים אהובים באמצעות Like, הורדת קבצים, ופונקציית חיפוש לפי שם הקובץ.

למנהל האתר קיימת האופציה לצפות בסטטיסטיקות שונות לגבי האתר כגון: מספר המבקרים, המשתמשים, מספר ההורדות.  
מנהל האתר יכול גם לערוך או למחוק קבצים ומשתמשים, בנוסף לכתיבת הודעה בדף הבית של האתר.

## **תאור קצר של המערכת:**

המערכת תאפשר למשתמשים לא רשומים לשמוע קבצי סאונד שמשתמשים העלו, להוריד סאונדים, לחפש סאונדים ולראות את כל הקבצים שהועלו על ידי משתמש מסוים.  
משתמשים רשומים יכולים גם להגיב על סאונדים, לעשות עליהם לייק, ולראות את כל הסאונדים שאהבו בדף אחד.  
המערכת תנהל את המשתמשים, על מה עשו לייק, ועל מה הגיבו, ואילו סאונדים הם העלו. בנוסף, המערכת תעקוב אחרי מספר ההשמעות, ההורדות, התגובות והלייקים של כל סאונד.

## **קהל יעד**

זמרים/יוצרים המחפשים פלטפורמה לשיתוף עבודותיהם, בין עם חבריהם ובין עם הקהל הרחב.

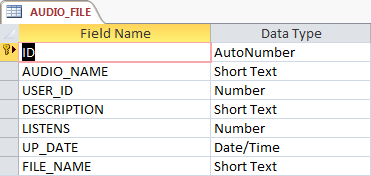
אנשים המחפשים שירים חדשים לשמוע.

אנשים הזקוקים לאחסון של קבצי אודיו מסיבות אלו או אחרות (האתר מאפשר הורדה של קבצים אלו כפי שהועלו).

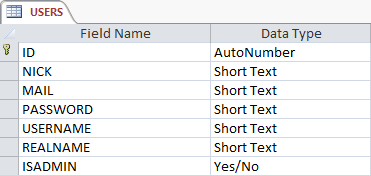
# **ניתוח המערכת – UML**

# **מבנה בסיס הנתונים**

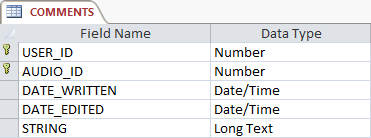
טבלת הסאונדים



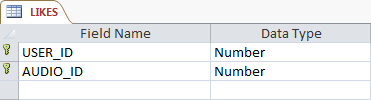
טבלת המשתמשים



טבלת תגובות



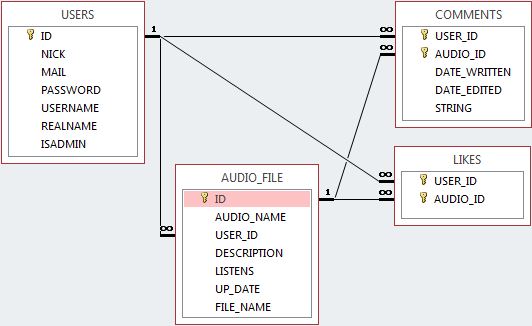
טבלת לייקים



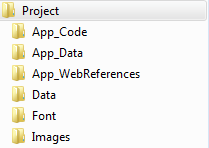
טבלה המכילה את ההודעה לדף הבית



## **קשרי גומלין (DSD)**



# **מבנה התיקיות**

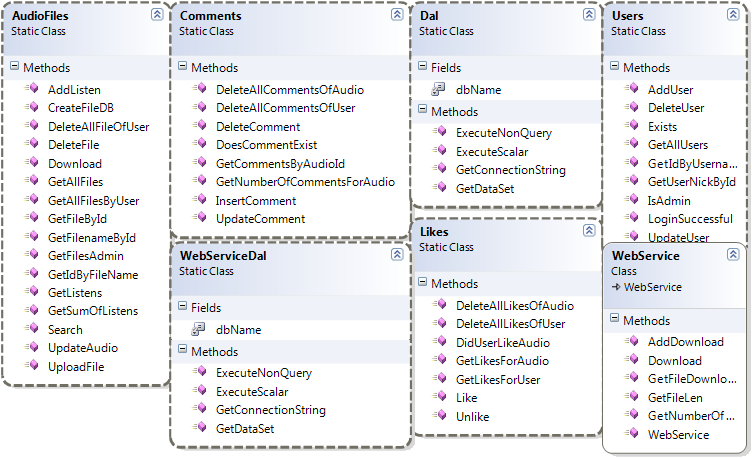


## מבנה התיקיות – הסבר

התיקיה Project: מכילה את אתר האינטרנט.

1. App\_Code: מחלקות לוגיות.
2. App\_Data: מסדי הנתונים.
3. App\_WebReferences: קישורים לשרותי רשת (Web Services)
4. Data: קבצי הסאונד.
5. Font: פונט האתר.
6. Images: תמונות לאתר.

# **המחלקות**

1. Dal – טיפול בגישה למסדי נתונים.
2. WebServiceDal – טיפול בגישה למסד הנתונים של הווב סרוויס.
3. AudioFiles – טיפול בקבצי אודיו.
4. Users – טיפול במשתמשים.
5. Comments – טיפול בתגובות.
6. Likes – טיפול בלייקים.

# **שרותי הרשת באתר**

שרותי רשת הנצרכים ממקור חיצוני

אין

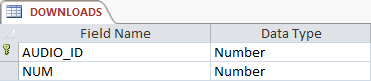
שרותי רשת שהאפליקציה מספקת

הורדת קבצי אודיו שהועלו באתר.  
קבלת אורך כל קובץ.

ספירת ההשמעות של כל הקבצים.

## **מבנה בסיס הנתונים של שירות הרשת**

טבלת מספר הורדות



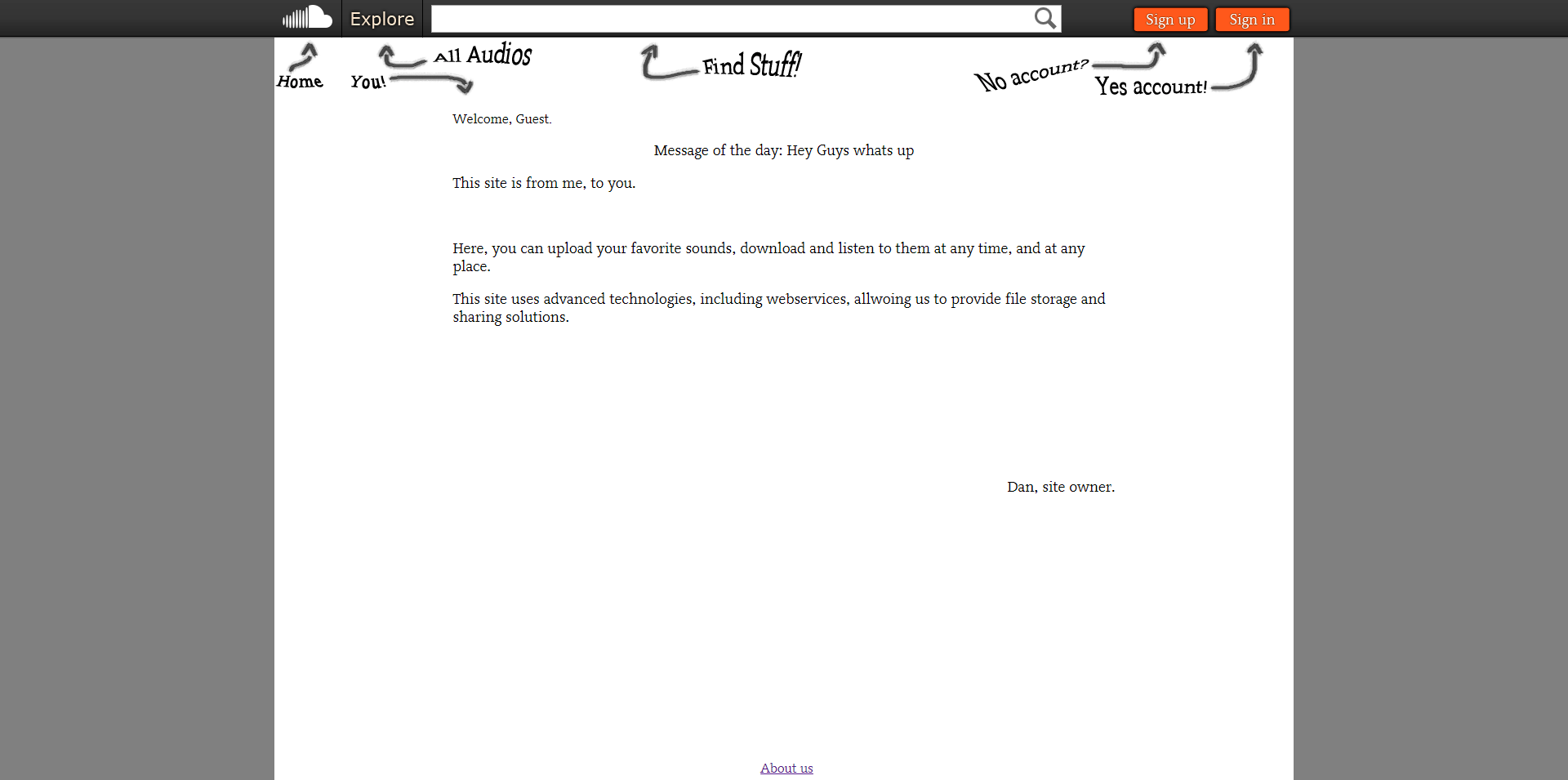
קשרי גומלין  
לא קיימים.

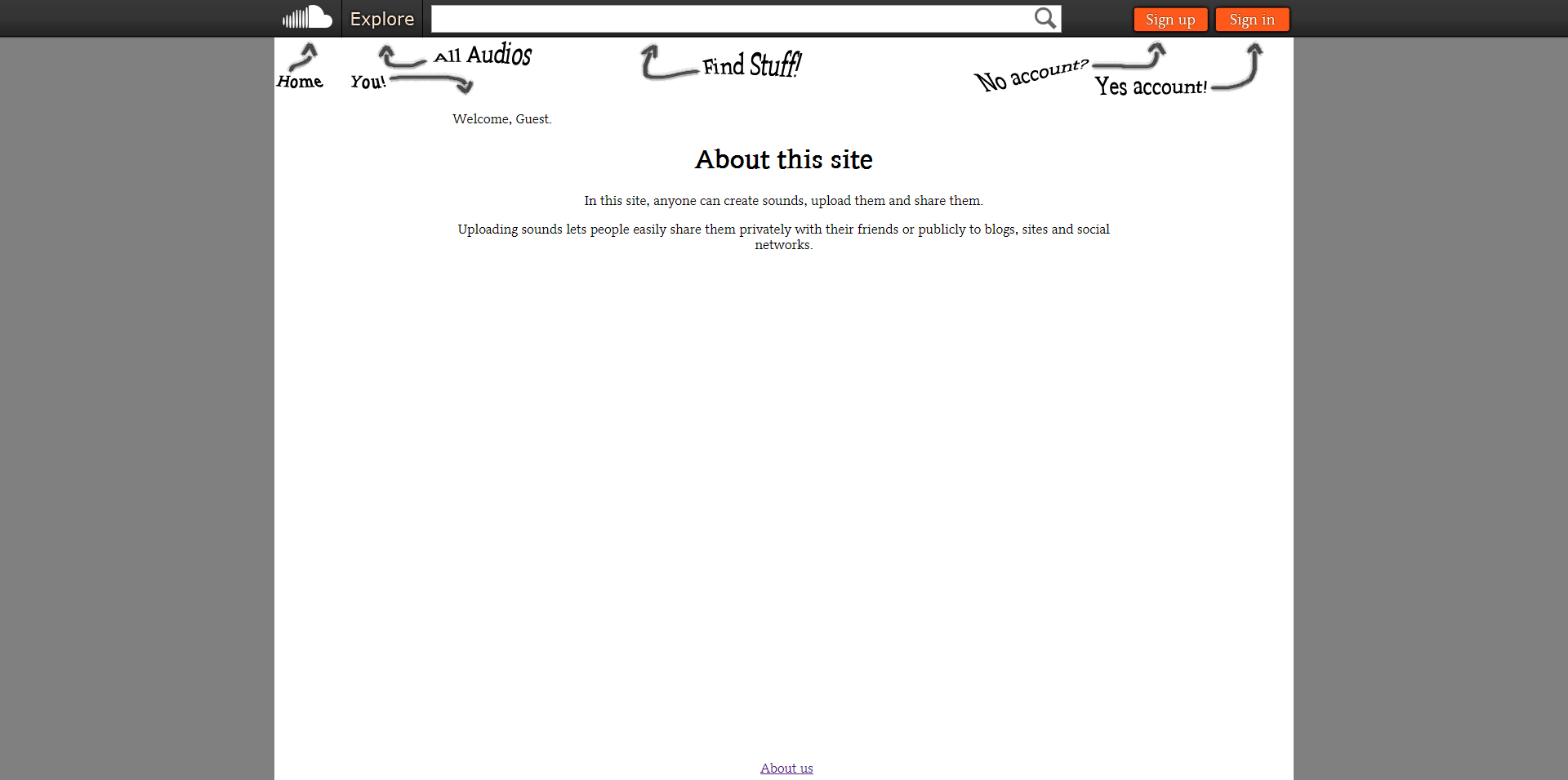
# **מפת האתר**

Green = Guest  
Orange = User  
Red = Admin

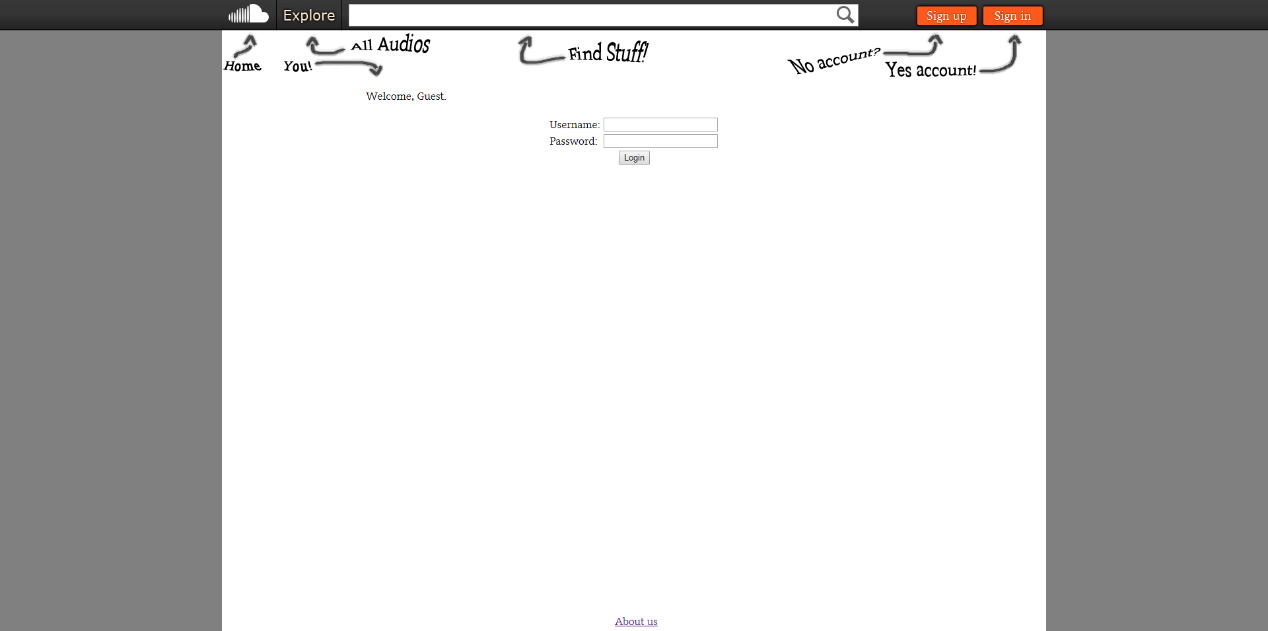
# **מדריך האפליקציה**

## **אורח**

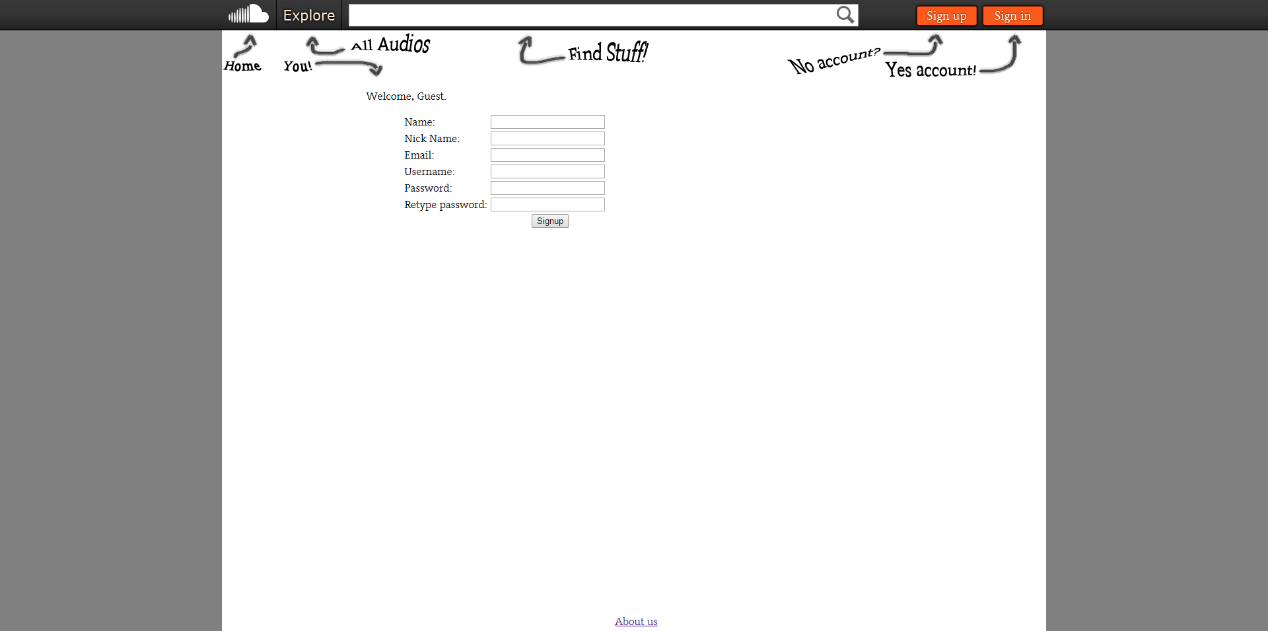
דף הבית  


אודות  


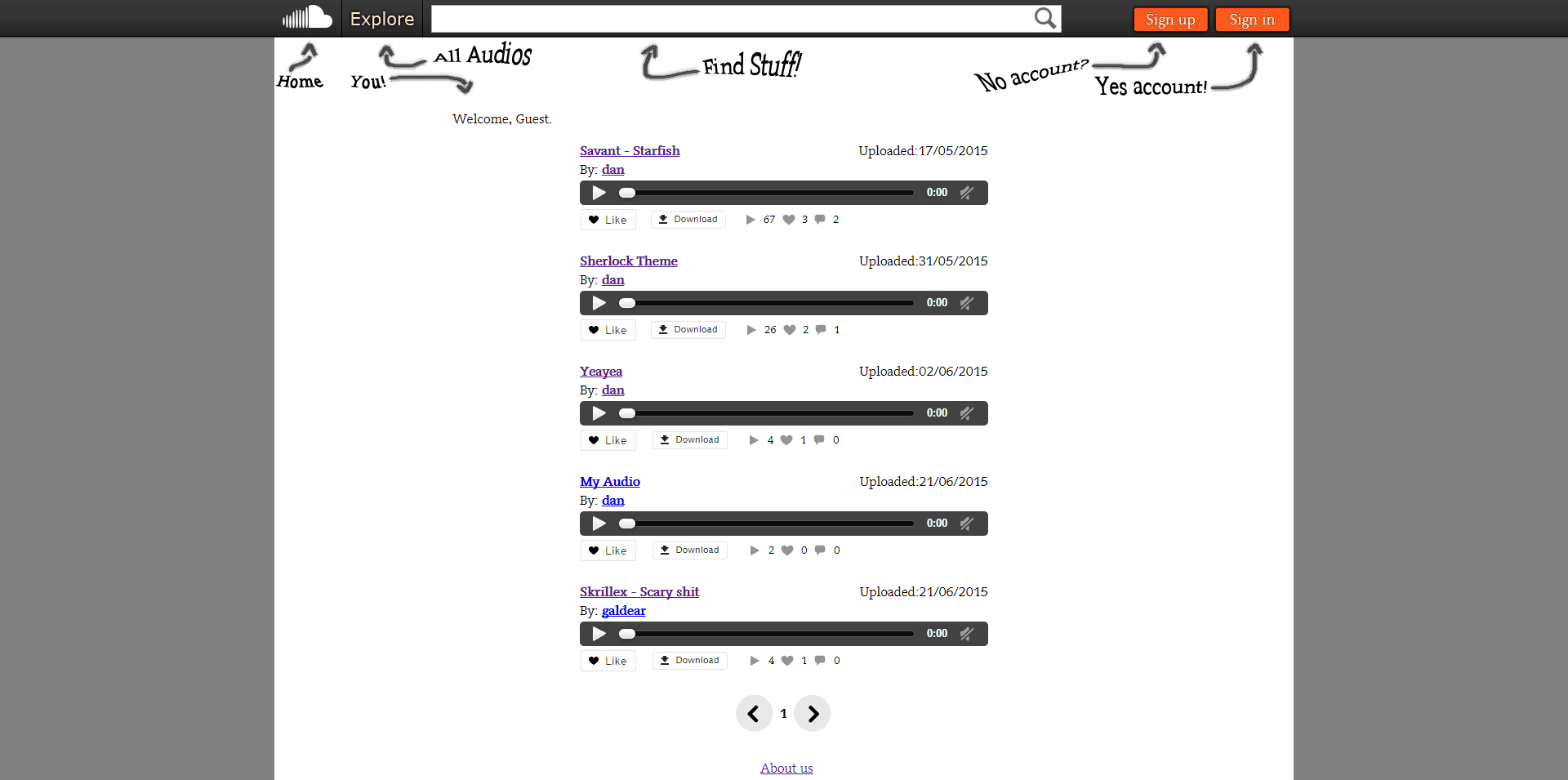
התחברות



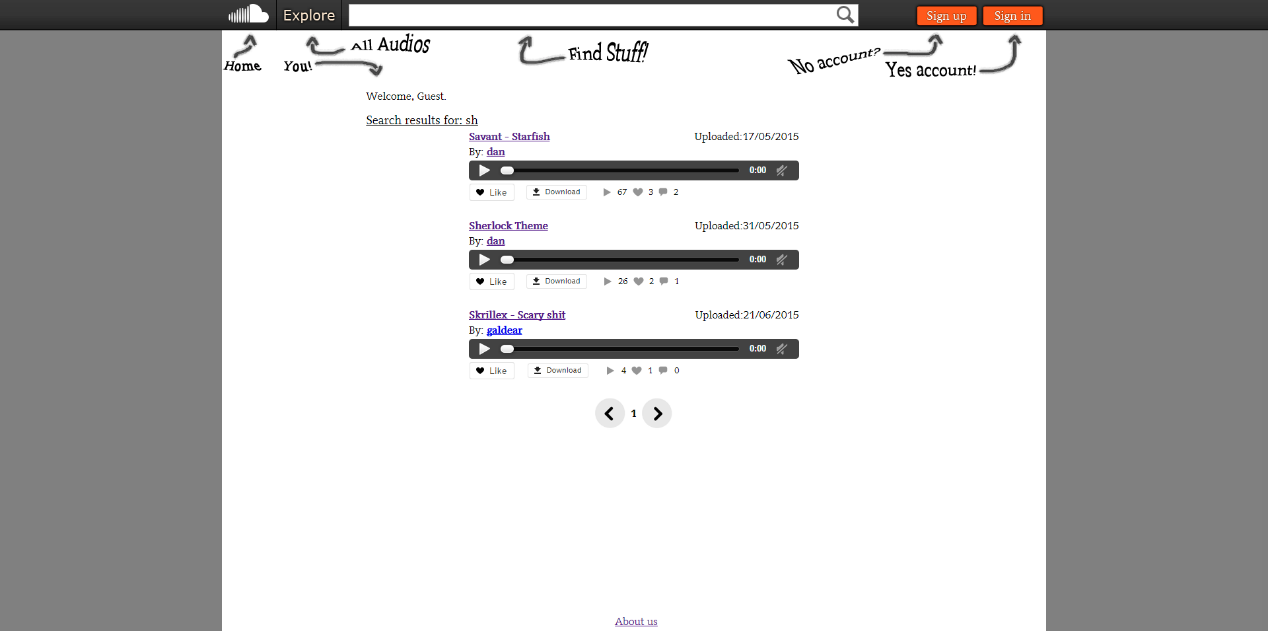
הרשמה



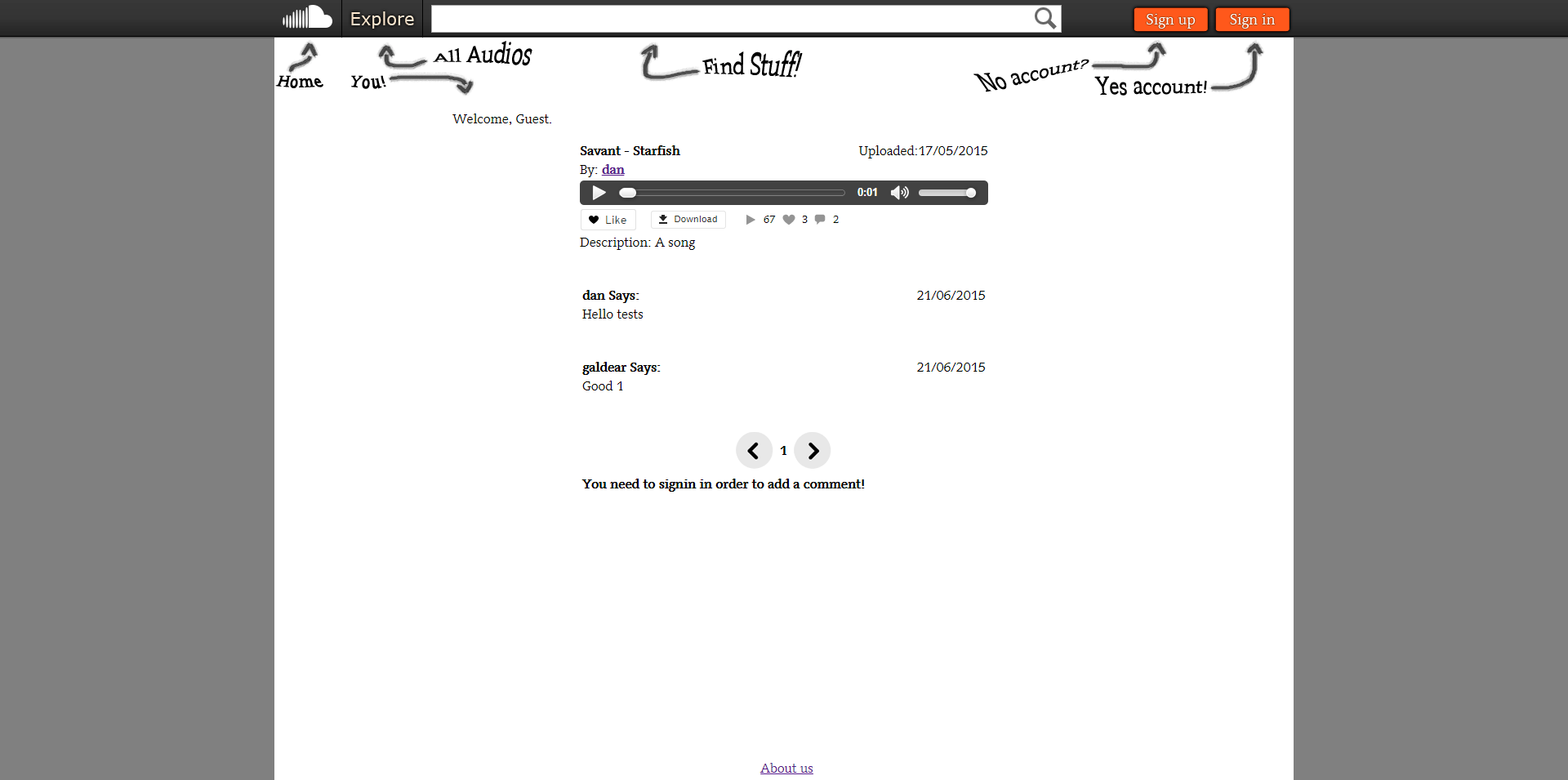
אקספלור (דף המכיל את כל הקבצים שהועלו)



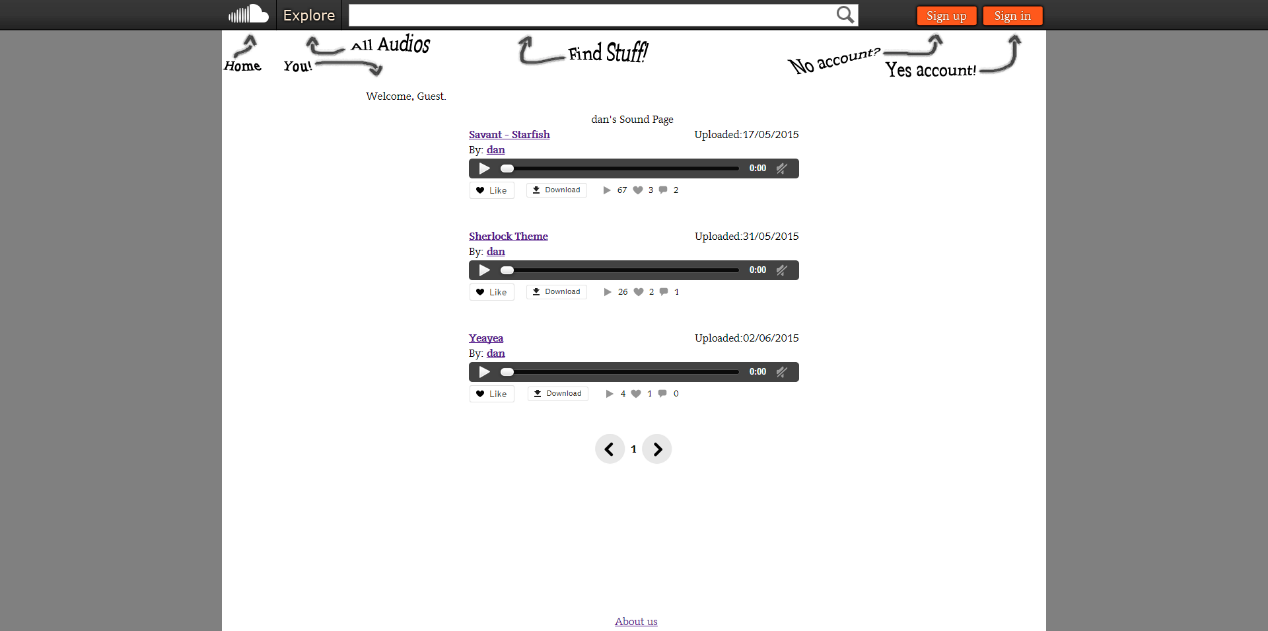
דף חיפוש (מגיעים אליו באמצעות הקלדת טקסט בתיבה למעלה ולחיצה על סמל החיפוש)



דף אודיו

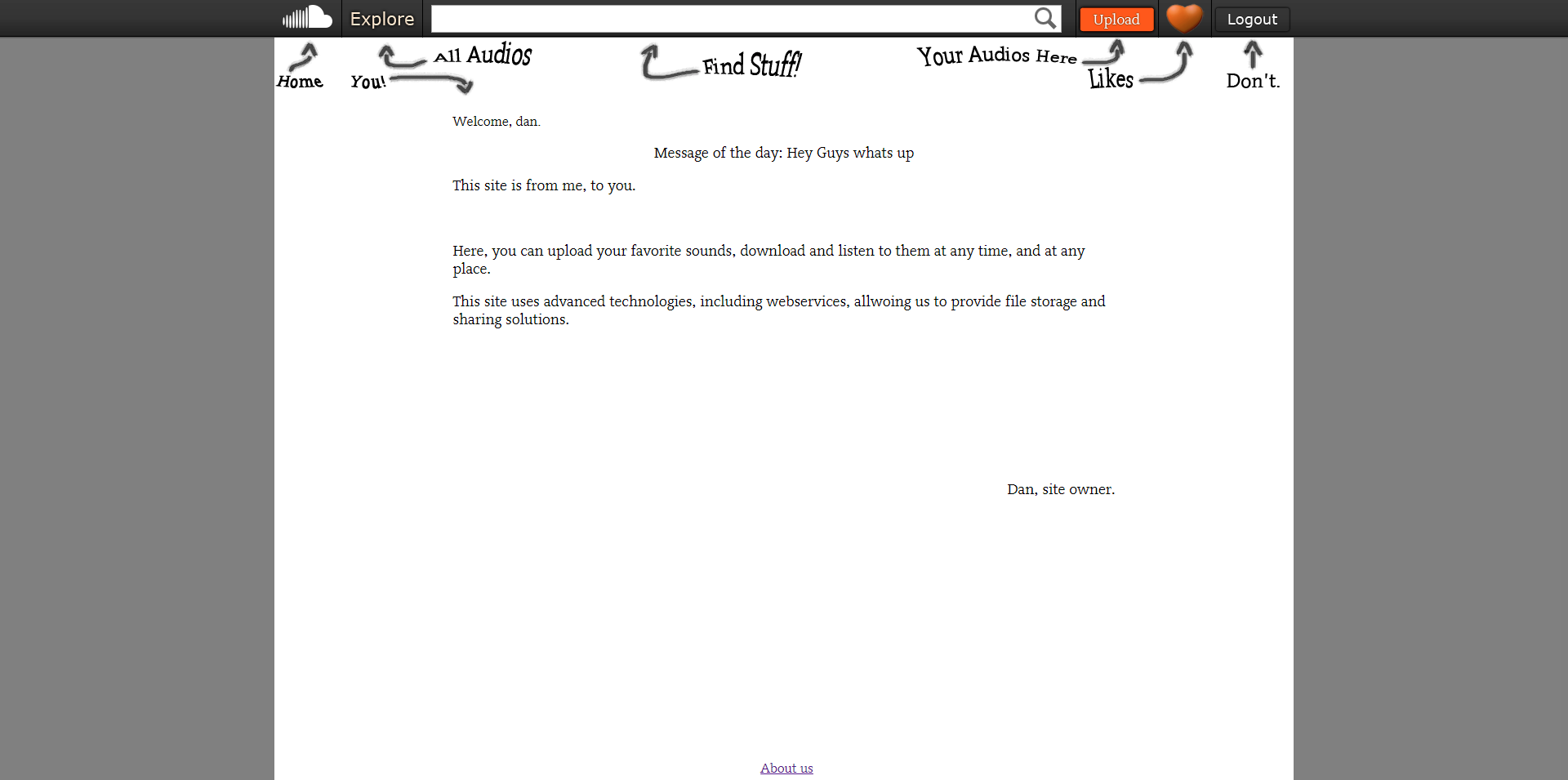


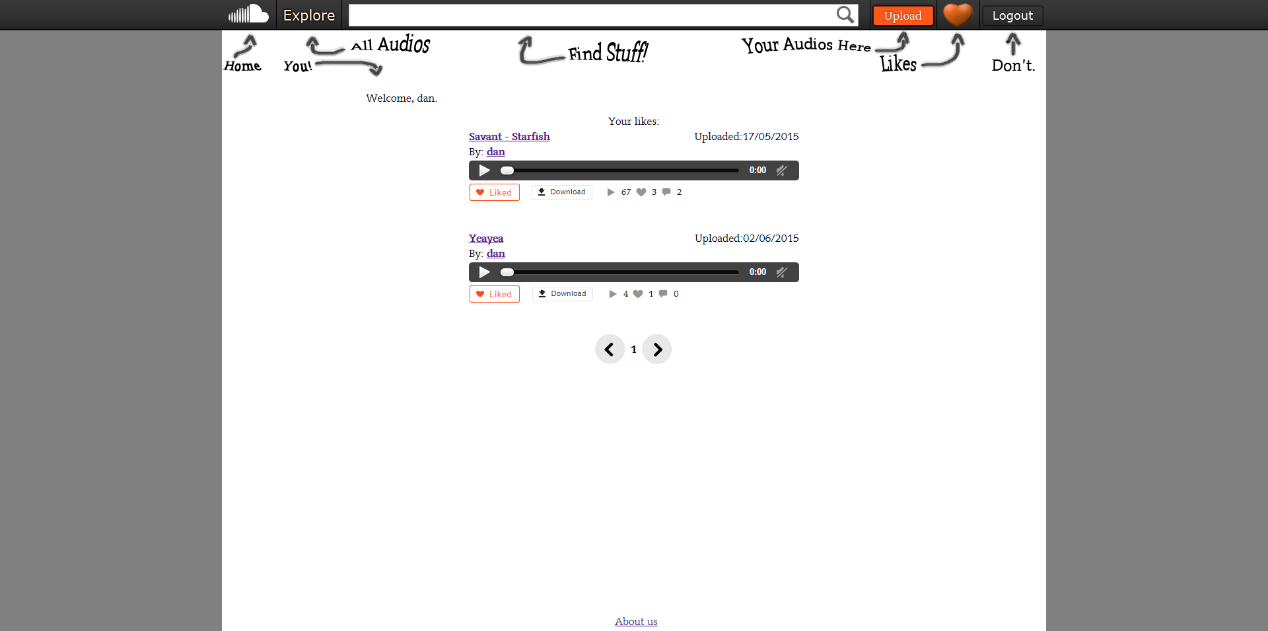
דף משתמש



## **משתמש רשום**

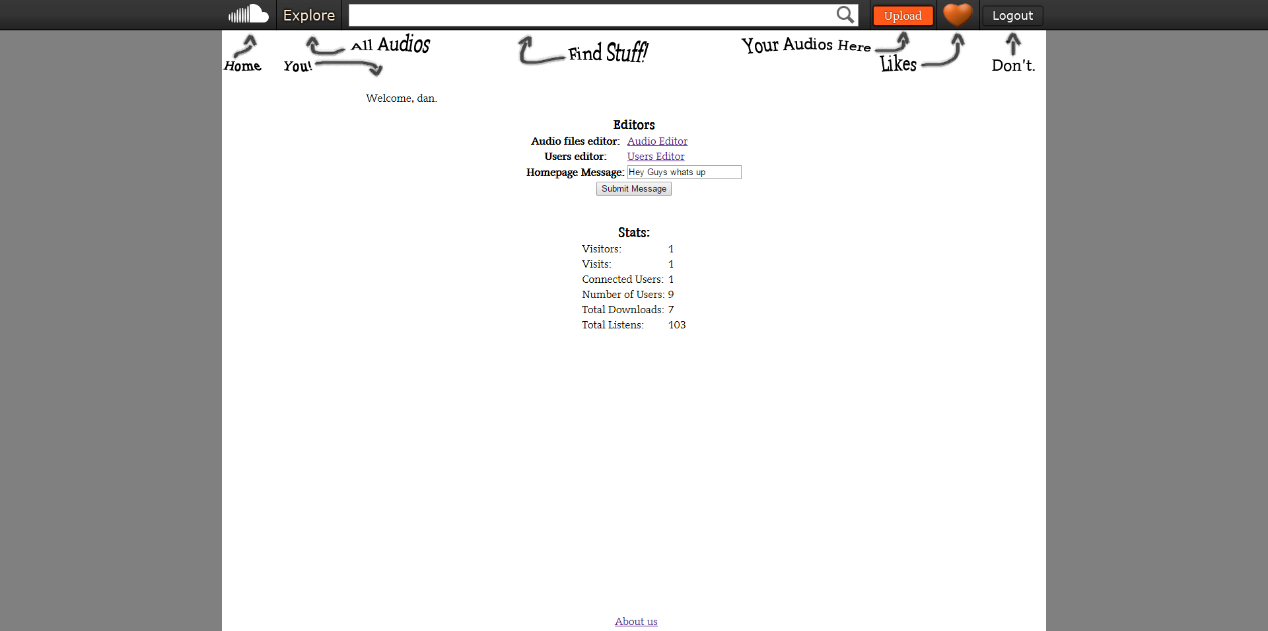
דף העלאת קבצים

****

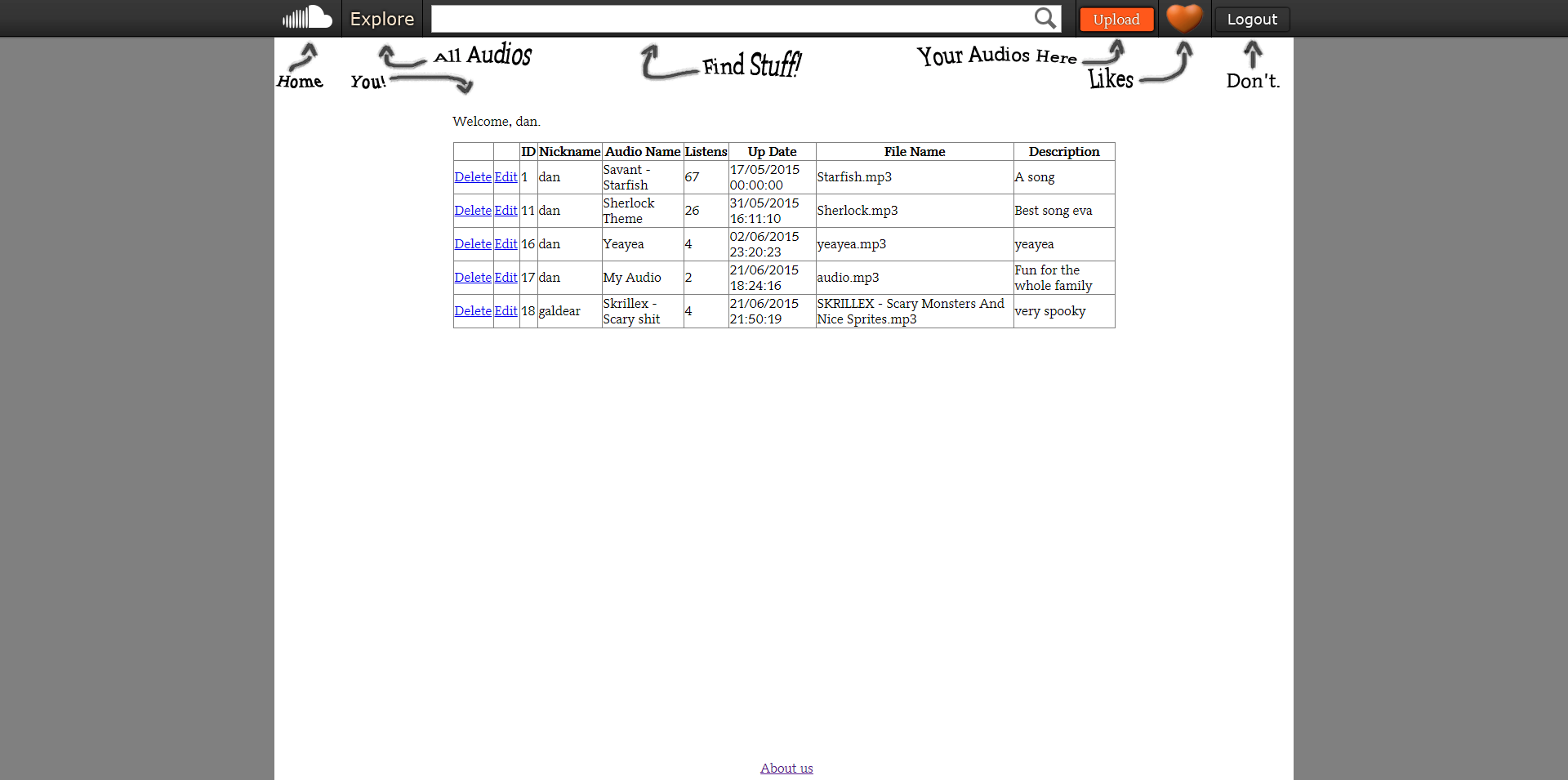
דף לייקים

## **אדמין**

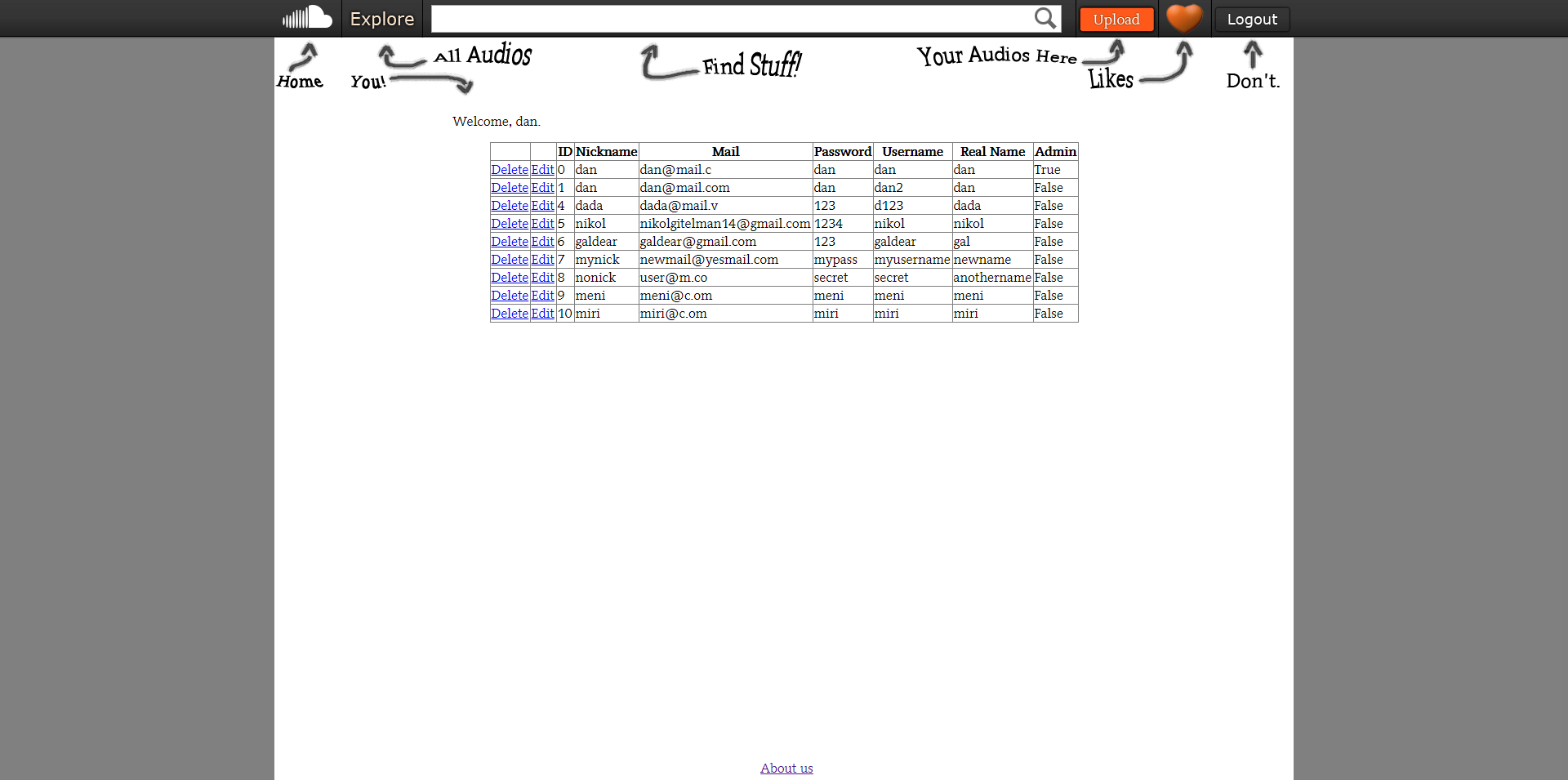
דף אדמין Admin.aspx



עורך פרטי קבצי אודיו



עורך משתמשים



# **דפי קוד**

## **מחלקות**

**AudioFiles.cs**public static class AudioFiles

{

//Returns all audio files for user page

public static DataSet GetAllFilesByUser(int userid)

{

string sql = string.Format("SELECT USERS.USERNAME,USERS.NICK,AUDIO\_FILE.ID, AUDIO\_FILE.AUDIO\_NAME, AUDIO\_FILE.DESCRIPTION, AUDIO\_FILE.LISTENS, AUDIO\_FILE.UP\_DATE FROM USERS, AUDIO\_FILE WHERE (((USERS.ID)=[AUDIO\_FILE].[USER\_ID]) AND USERS.ID = {0})", userid);

return Dal.GetDataSet(sql);

}

//Returns all audio files for explore view

public static DataSet GetAllFiles()

{

string sql = string.Format("SELECT USERS.USERNAME,USERS.NICK,AUDIO\_FILE.ID, AUDIO\_FILE.AUDIO\_NAME, AUDIO\_FILE.LISTENS, AUDIO\_FILE.UP\_DATE FROM USERS, AUDIO\_FILE WHERE (((USERS.ID)=[AUDIO\_FILE].[USER\_ID]))");

return Dal.GetDataSet(sql);

}

//Returns audios with name that fits search string

public static DataSet Search(string SearchString)

{

string sql = string.Format("SELECT USERS.USERNAME,USERS.NICK,AUDIO\_FILE.ID, AUDIO\_FILE.AUDIO\_NAME, AUDIO\_FILE.DESCRIPTION, AUDIO\_FILE.LISTENS, AUDIO\_FILE.UP\_DATE FROM USERS, AUDIO\_FILE WHERE (((USERS.ID)=[AUDIO\_FILE].[USER\_ID]) AND AUDIO\_NAME LIKE '%{0}%')", SearchString);

return Dal.GetDataSet(sql);

}

//Upload method

public static string UploadFile(FileUpload FileUpload1,string AudioName,string AudioDesc,string UserId)

{

string label1="";

if (CreateFileDB(AudioName,AudioDesc,UserId,FileUpload1.FileName) != 0)

{

try

{

FileUpload1.SaveAs(HttpContext.Current.Server.MapPath("~/Data/") +GetIdByFileName(FileUpload1.FileName).ToString()+".mp3");

label1 = "File name: " +

AudioName + "<br>" +

(int)(FileUpload1.PostedFile.ContentLength)/1024 + " Kb<br>" +

"Content type: " +

FileUpload1.PostedFile.ContentType;

}

catch (Exception ex)

{

label1 = "ERROR: " + ex.Message.ToString();

}

}

return label1;

}

//Adds file to DB then returns id

public static int CreateFileDB(string AudioName, string AudioDesc, string UserId, string FileName)

{

string sql = string.Format("INSERT INTO AUDIO\_FILE ([AUDIO\_NAME],[USER\_ID],[DESCRIPTION],[LISTENS],[UP\_DATE],[FILE\_NAME]) VALUES ('{0}','{1}','{2}',0,'{3}','{4}')",

AudioName, UserId, AudioDesc, System.DateTime.Now , FileName);

return (int)Dal.ExecuteNonQuery(sql);

}

//Retruns id by filename

public static int GetIdByFileName(string filename)

{

string sql = string.Format("SELECT \* FROM AUDIO\_FILE WHERE FILE\_NAME='{0}'",filename);

return (int)Dal.ExecuteScalar(sql);

}

//Retruns filename by id

public static string GetFilenameById(int id)

{

string sql = string.Format("SELECT FILE\_NAME FROM AUDIO\_FILE WHERE ID={0}", id);

return Dal.ExecuteScalar(sql).ToString();

}

//Retruns num of listens for id

public static int GetListens(int audio\_id)

{

string sql = string.Format("SELECT LISTENS FROM AUDIO\_FILE WHERE (ID = {0})", audio\_id);

return (int)Dal.ExecuteScalar(sql);

}

//Adds a listen to file id

public static void AddListen(int audio\_id)

{

string sql = string.Format("UPDATE AUDIO\_FILE SET LISTENS={0} WHERE (ID = {1})",GetListens(audio\_id)+1 ,audio\_id);

Dal.ExecuteNonQuery(sql);

}

//Retruns a file by his id

public static DataSet GetFileById(int id)

{

string sql = string.Format("SELECT USERS.NICK,USERS.USERNAME,AUDIO\_FILE.ID, AUDIO\_FILE.AUDIO\_NAME, AUDIO\_FILE.DESCRIPTION, AUDIO\_FILE.LISTENS, AUDIO\_FILE.UP\_DATE FROM USERS, AUDIO\_FILE WHERE (((USERS.ID)=[AUDIO\_FILE].[USER\_ID]) AND AUDIO\_FILE.ID={0})", id);

return Dal.GetDataSet(sql);

}

//Deletes file by his id

public static bool DeleteFile(int id)

{

Comments.DeleteAllCommentsOfAudio(id);

Likes.DeleteAllLikesOfAudio(id);

string sql = string.Format("DELETE FROM AUDIO\_FILE Where id={0}", id);

Dal.ExecuteNonQuery(sql);

if (File.Exists(HttpContext.Current.Server.MapPath("~/Data/") + id + ".mp3"))

{

File.Delete(HttpContext.Current.Server.MapPath("~/Data/") + id + ".mp3");

}

return true;

}  
 //Retruns AudioFile data for admin page

public static DataSet GetFilesAdmin()

{

string sql = string.Format("SELECT USERS.NICK,AUDIO\_FILE.ID, AUDIO\_FILE.AUDIO\_NAME, AUDIO\_FILE.LISTENS, AUDIO\_FILE.UP\_DATE,AUDIO\_FILE.DESCRIPTION,AUDIO\_FILE.FILE\_NAME FROM USERS, AUDIO\_FILE WHERE (((USERS.ID)=[AUDIO\_FILE].[USER\_ID]))");

return Dal.GetDataSet(sql);

}

//Retruns File in Byte[] form

public static Byte[] Download(int id)

{

ServiceReference1.WebServiceSoapClient ws = new ServiceReference1.WebServiceSoapClient();

Byte[] file = ws.Download(id);

return file;

}

//Updates AudioFile details by his id

public static void UpdateAudio(int id, string audio\_name,int listens, DateTime UpDate, string filename, string desc)

{

string sql = string.Format("UPDATE AUDIO\_FILE SET AUDIO\_NAME='{0}',LISTENS={1},UP\_DATE='{2}',FILE\_NAME='{3}',DESCRIPTION='{4}' WHERE (ID={5})",audio\_name,listens,UpDate,filename,desc, id);

Dal.ExecuteNonQuery(sql);

}

//Retruns sum of listens

public static int GetSumOfListens()

{

string sql = string.Format("SELECT SUM(LISTENS) FROM AUDIO\_FILE");

return int.Parse(Dal.ExecuteScalar(sql).ToString());

}

//Deletes all files that a user has uploaded

public static void DeleteAllFileOfUser(int user\_id)

{

DataSet ds = GetAllFilesByUser(user\_id);

int id;

int num = ds.Tables[0].Rows.Count;

for (int i = 0; i < num; i++)

{

id=ds.Tables[0].Rows[i].Field<int>("ID");

DeleteFile(id);

}

}

}

**Comments.cs**public static class Comments

{

//Gets number of comments on audio

public static int GetNumberOfCommentsForAudio(int audio\_id)

{

string sql = string.Format("SELECT COUNT(USER\_ID) FROM COMMENTS WHERE AUDIO\_ID ={0}", audio\_id);

return (int)Dal.ExecuteScalar(sql);

}

//Deletes a comment

public static bool DeleteComment(int user\_id, int audio\_id)

{

try

{

string sql = string.Format("DELETE FROM COMMENTS WHERE AUDIO\_ID ={0} AND USER\_ID={1}", audio\_id, user\_id);

Dal.ExecuteNonQuery(sql);

}

catch (Exception e)

{

return false;

}

return true;

}

//Deletes all comments by audio id

public static bool DeleteAllCommentsOfAudio(int audio\_id)

{

try

{

string sql = string.Format("DELETE FROM COMMENTS WHERE AUDIO\_ID ={0}", audio\_id);

Dal.ExecuteNonQuery(sql);

}

catch (Exception e)

{

return false;

}

return true;

}

//Deletes all comments by user id

public static bool DeleteAllCommentsOfUser(int user\_id)

{

try

{

string sql = string.Format("DELETE FROM COMMENTS WHERE USER\_ID ={0}", user\_id);

Dal.ExecuteNonQuery(sql);

}

catch (Exception e)

{

return false;

}

return true;

}

//Gets all comments for audio by it's id

public static DataSet GetCommentsByAudioId(int id)

{

string sql = string.Format("SELECT USERS.NICK, COMMENTS.USER\_ID, COMMENTS.DATE\_WRITTEN, COMMENTS.STRING FROM USERS,COMMENTS WHERE (((USERS.ID)=[COMMENTS].[USER\_ID]) AND (COMMENTS.AUDIO\_ID={0}))",id);

return Dal.GetDataSet(sql);

}

//Adds a comment

public static void InsertComment(int user\_id,int audio\_id, DateTime date\_w, string comment)

{

string sql = string.Format("INSERT INTO COMMENTS VALUES ({0},{1},#{2}#,#{3}#,'{4}')", user\_id, audio\_id, date\_w, date\_w, comment);

Dal.ExecuteNonQuery(sql);

}

//Updates comment

public static void UpdateComment(int user\_id, int audio\_id, DateTime date\_w, string comment)

{

string sql = string.Format("UPDATE COMMENTS SET [STRING]='{0}', DATE\_WRITTEN='{1}' WHERE (USER\_ID={2}) AND (AUDIO\_ID={3})",comment,date\_w,user\_id,audio\_id);

Dal.ExecuteNonQuery(sql);

}

//Checks if comment exists

public static bool DoesCommentExist(int user\_id, int audio\_id)

{

string sql = string.Format("SELECT \* FROM COMMENTS WHERE (USER\_ID={0} AND AUDIO\_ID={1})", user\_id,audio\_id);

return Dal.ExecuteScalar(sql)!=null;

}

}

**Dal.cs**static public class Dal

{

const string dbName = "DB.accdb";

public static string GetConnectionString()

{

return "Provider=Microsoft.ACE.OLEDB.12.0; Data Source=" + HttpContext.Current.Server.MapPath("~/App\_Data/") + dbName+"; Persist Security Info=True";

}

//מחרוזת SQL

//מספר שורות ששונו

static public int ExecuteNonQuery(string sql)

{

int numOfUpdateRows;

OleDbConnection con = null;

OleDbCommand cmd = null;

try

{

con = new OleDbConnection(GetConnectionString());

cmd = new OleDbCommand(sql, con);

con.Open();

numOfUpdateRows = cmd.ExecuteNonQuery();

}

catch (Exception e)

{

numOfUpdateRows=0;

}

finally

{

con.Close();

}

return numOfUpdateRows;

}

// משפט SQL

//מוחזר העצם עליו בוצעה הפעולה

static public object ExecuteScalar(string sql)

{

OleDbConnection con = null;

OleDbCommand cmd = null;

object val = null;

try

{

con = new OleDbConnection(GetConnectionString());

cmd = new OleDbCommand(sql, con);

con.Open();

val = cmd.ExecuteScalar();

}

catch (Exception e)

{

val = null;

}

finally

{

con.Close();

}

return val;

}

// // משפט SQL

// הפעולה מחזירה אוסף טבלאות בהתאם למשפט SQL

static public DataSet GetDataSet(string sql)

{

OleDbConnection con = null;

DataSet ds = null;

try

{

con = new OleDbConnection(GetConnectionString());

OleDbDataAdapter da = new OleDbDataAdapter(sql, con);

ds = new DataSet();

da.Fill(ds);

}

catch (Exception e)

{

ds = null;

}

return ds;

}

}

**Likes.cs**public static class Likes

{

//returns amount of likes

public static int GetLikesForAudio(int audio\_id)

{

string sql = string.Format("SELECT COUNT(USER\_ID) FROM LIKES WHERE AUDIO\_ID ={0}", audio\_id);

return (int)Dal.ExecuteScalar(sql);

}

//returns true if use liked audio, else false

public static bool DidUserLikeAudio(int user\_id, int audio\_id)

{

string sql = string.Format("SELECT USER\_ID FROM LIKES WHERE AUDIO\_ID ={0} AND USER\_ID={1}", audio\_id, user\_id);

return Dal.ExecuteScalar(sql) != null;

}

//Likes an audio

public static void Like(int user\_id, int audio\_id)

{

string sql = string.Format("INSERT INTO LIKES ([USER\_ID],[AUDIO\_ID]) VALUES ({0},{1}) ", user\_id, audio\_id);

Dal.ExecuteNonQuery(sql);

}

//unikes an audio

public static bool Unlike(int user\_id, int audio\_id)

{

string sql = string.Format("DELETE FROM LIKES WHERE AUDIO\_ID ={0} AND USER\_ID={1}", audio\_id, user\_id);

Dal.ExecuteNonQuery(sql);

return true;

}

// Gets user liked songs

public static DataSet GetLikesForUser(int user\_id)

{

string sql = string.Format("SELECT AUDIO\_FILE.AUDIO\_NAME, AUDIO\_FILE.LISTENS, AUDIO\_FILE.UP\_DATE, USERS.NICK, USERS.USERNAME, AUDIO\_FILE.ID FROM USERS INNER JOIN (AUDIO\_FILE INNER JOIN LIKES ON AUDIO\_FILE.ID = LIKES.AUDIO\_ID) ON USERS.ID = LIKES.USER\_ID WHERE (((LIKES.USER\_ID)={0}));", user\_id);

return Dal.GetDataSet(sql);

}

// Deletes all likes related to audio. used in song deletion.

public static void DeleteAllLikesOfAudio(int audio\_id)

{

string sql = string.Format("DELETE FROM LIKES WHERE AUDIO\_ID ={0}", audio\_id);

Dal.ExecuteNonQuery(sql);

}

// Deletes all likes related to user. used in user deletion.

public static void DeleteAllLikesOfUser(int user\_id)

{

string sql = string.Format("DELETE FROM LIKES WHERE USER\_ID ={0}", user\_id);

Dal.ExecuteNonQuery(sql);

}

}

**Users.cs**static public class Users

{

// מחזיר 1 אם הוסיף משתמש אחרת מחזיר את הבעיה

public static string AddUser(string nick,string mail, string password,string username, string realname)

{

string exists = Exists(username, mail);

if (exists=="")

{

string sql = string.Format("INSERT INTO USERS ([NICK],[MAIL],[PASSWORD],[USERNAME],[REALNAME],[ISADMIN]) VALUES ('{0}','{1}','{2}','{3}','{4}', 0)",

nick, mail, password, username, realname);

return Dal.ExecuteNonQuery(sql).ToString();

}

return exists;

}

// בודק אם קיים כבר מייל או שם משתמש

public static String Exists(string username,string mail)

{

string error="";

string sql = string.Format("SELECT USERNAME FROM USERS WHERE USERNAME='{0}'", username);

if (Dal.ExecuteScalar(sql) != null)

error += "username in use ";

sql = string.Format("SELECT MAIL FROM USERS WHERE MAIL='{0}'", mail);

if (Dal.ExecuteScalar(sql) != null)

error += "mail in use";

return error;

}

//deletes a user and all related songs, comments and likes.

public static bool DeleteUser(int id)

{

try

{

AudioFiles.DeleteAllFileOfUser(id);

Comments.DeleteAllCommentsOfUser(id);

Likes.DeleteAllLikesOfUser(id);

string sql = string.Format("DELETE FROM USERS WHERE ID={0}", id);

Dal.ExecuteNonQuery(sql);

}

catch (Exception e)

{

return false;

}

return true;

}

//מחזיר אמת אם הצליח להתחבר, אחרת שקר

public static bool LoginSuccessful(string username, string pass)

{

string sql = string.Format("SELECT USERNAME FROM USERS WHERE PASSWORD= '{0}' AND USERNAME='{1}'", pass, username);

return Dal.ExecuteScalar(sql) != null;

}

//returns true if user is admin, else false

public static bool IsAdmin(int id)

{

string sql = string.Format("SELECT ISADMIN FROM USERS WHERE ID={0}",id);

return (bool)Dal.ExecuteScalar(sql);

}

//returns user nick by his id

public static string GetUserNickById(int id) //asks for id

{

string sql = string.Format("SELECT NICK FROM USERS WHERE ID={0}", id);

return Dal.ExecuteScalar(sql).ToString();

}

//returns user id by his username

public static int GetIdByUsername(string username)

{

string sql = string.Format("SELECT ID FROM USERS WHERE USERNAME='{0}'", username);

return (int)Dal.ExecuteScalar(sql);

}

//returns all users

public static DataSet GetAllUsers()

{

string sql = string.Format("SELECT \* FROM USERS");

return Dal.GetDataSet(sql);

}

//updates user info

public static void UpdateUser(int id, string nick, string pass, string realname)

{

string sql = string.Format("UPDATE USERS SET NICK='{0}',PASSWORD='{1}',REALNAME='{2}' WHERE (ID={3})",nick,pass,realname,id);

Dal.ExecuteNonQuery(sql);

}

}

## **WebService**

**WebService.cs**public class WebService : System.Web.Services.WebService {

public WebService () {

//Uncomment the following line if using designed components

//InitializeComponent();

}

//Sends file from ws to server in Byte[]

[WebMethod]

public Byte[] Download(int audio\_id)

{

string filepath = HttpContext.Current.Server.MapPath("~/Data/") + audio\_id + ".mp3";

if (File.Exists(filepath))

{

FileStream objfilestream = new FileStream(filepath, FileMode.Open, FileAccess.Read);

int len = (int)objfilestream.Length;

Byte[] file = new Byte[len];

objfilestream.Read(file, 0, len);

objfilestream.Close();

AddDownload(audio\_id);

return file;

}

else

return new Byte[0];

}

//returns file length

[WebMethod]

public int GetFileLen(int id)

{

string filepath = "G:\\Project\\Data\\" + id + ".mp3";

FileStream objfilestream = new FileStream(filepath, FileMode.Open, FileAccess.Read);

int len = (int)objfilestream.Length;

objfilestream.Close();

return len;

}

//returns number of downloads for file id

[WebMethod]

public int GetFileDownloads(int audio\_id)

{

string sql = string.Format("SELECT NUM FROM DOWNLOADS WHERE AUDIO\_ID={0}", audio\_id);

if (WebServiceDal.ExecuteScalar(sql) == null)

{

sql = string.Format("INSERT INTO DOWNLOADS VALUES ({0},0)", audio\_id);

WebServiceDal.ExecuteNonQuery(sql);

return 0;

}

else

return (int)WebServiceDal.ExecuteScalar(sql);

}

//returns sum of downloads, used in admin stat page

[WebMethod]

public int GetNumberOfDownloads()

{

string sql = string.Format("SELECT SUM(NUM) FROM DOWNLOADS");

return int.Parse(WebServiceDal.ExecuteScalar(sql).ToString());

}

//adds download in db for id

public void AddDownload(int audio\_id)

{

string sql = string.Format("UPDATE DOWNLOADS SET NUM={0} WHERE AUDIO\_ID={1}",GetFileDownloads(audio\_id)+1,audio\_id);

WebServiceDal.ExecuteNonQuery(sql);

}

}

## **דפים**

**Admin.aspx.cs**public partial class Admin : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Admin"] == null || !(bool)Session["Admin"])

Response.Redirect("Home.aspx");

if (!IsPostBack)

{

TextBoxMessage.Text = Dal.ExecuteScalar("SELECT MESSAGE FROM SITE\_STRINGS").ToString();

}

}

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

{

Dal.ExecuteNonQuery(string.Format("UPDATE SITE\_STRINGS SET MESSAGE='{0}'", @TextBoxMessage.Text.ToString()));

}

}

**AdminAudio.aspx.cs**public partial class AudioAdmin : System.Web.UI.Page

{

//Fires on page load

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Admin"] == null || !(bool)Session["Admin"])

Response.Redirect("Home.aspx");

if (!IsPostBack)

{

Refresh();

}

}

//Fires on updating a row

protected void GridView1\_RowUpdating(object sender, GridViewUpdateEventArgs e)

{

int id = int.Parse((GridView1.Rows[e.RowIndex].Cells[2]).Text);

string audio\_name = ((TextBox)(GridView1.Rows[e.RowIndex].Cells[4].Controls[0])).Text;

int listens = int.Parse(((TextBox)(GridView1.Rows[e.RowIndex].Cells[5].Controls[0])).Text);

DateTime UpDate = DateTime.Parse(((TextBox)(GridView1.Rows[e.RowIndex].Cells[6].Controls[0])).Text);

string filename = ((TextBox)(GridView1.Rows[e.RowIndex].Cells[7].Controls[0])).Text;

string desc = ((TextBox)(GridView1.Rows[e.RowIndex].Cells[8].Controls[0])).Text;

GridView1.EditIndex = -1;

AudioFiles.UpdateAudio(id, audio\_name, listens, UpDate, filename, desc);

Refresh();

}

//Fires on deleting a row

protected void GridView1\_RowDeleting(object sender, GridViewDeleteEventArgs e)

{

int rowIndex = Convert.ToInt32(e.RowIndex);

int id = int.Parse(GridView1.Rows[rowIndex].Cells[2].Text);

AudioFiles.DeleteFile(id);

Refresh();

}

//Fires when clicking editing button

protected void GridView1\_RowEditing(object sender, GridViewEditEventArgs e)

{

GridView1.EditIndex = e.NewEditIndex;

Refresh();

}

//Fires when clicking cancel button

protected void GridView1\_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)

{

GridView1.EditIndex = -1;

Refresh();

}

//Fires when changing page

protected void GridView1\_PageIndexChanging(object sender, GridViewPageEventArgs e)

{

GridView1.PageIndex = e.NewPageIndex;

Refresh();

}

//Refreshes the Grid

protected void Refresh()

{

GridView1.DataSource = AudioFiles.GetFilesAdmin();

GridView1.DataBind();

}

}

**AdminUsers.aspx.cs**

public partial class AdminUsers : System.Web.UI.Page

{

//Fires on page load

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Admin"] == null || !(bool)Session["Admin"])

Response.Redirect("Home.aspx");

if (!IsPostBack)

{

Refresh();

}

}

------------------------------------------------------------🡪

//Fires when updating a row

protected void GridView1\_RowUpdating(object sender, GridViewUpdateEventArgs e)

{

int id = int.Parse((GridView1.Rows[e.RowIndex].Cells[2]).Text);

string nick = ((TextBox)(GridView1.Rows[e.RowIndex].Cells[3].Controls[0])).Text;

string password = ((TextBox)(GridView1.Rows[e.RowIndex].Cells[5].Controls[0])).Text;

string realname = ((TextBox)(GridView1.Rows[e.RowIndex].Cells[7].Controls[0])).Text;

GridView1.EditIndex = -1;

Users.UpdateUser(id,nick,password,realname);

Refresh();

}

//Fires when deleting a row

protected void GridView1\_RowDeleting(object sender, GridViewDeleteEventArgs e)

{

int rowIndex = Convert.ToInt32(e.RowIndex);

int id = int.Parse(GridView1.Rows[rowIndex].Cells[2].Text);

if (Session["ID"].ToString() != id.ToString() && !Users.IsAdmin(id))

{

Users.DeleteUser(id);

Refresh();

}

else

LabelError.Text = "Can't delete yourself or admin.";

}

//Fires when clicking update button

protected void GridView1\_RowEditing(object sender, GridViewEditEventArgs e)

{

GridView1.EditIndex = e.NewEditIndex;

Refresh();

}

//Fires when clicking cancel button

protected void GridView1\_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)

{

GridView1.EditIndex = -1;

Refresh();

}

//Fires when changing page

protected void GridView1\_PageIndexChanging(object sender, GridViewPageEventArgs e)

{

GridView1.PageIndex = e.NewPageIndex;

Refresh();

}

//Refreshes the Grid

protected void Refresh()

{

GridView1.DataSource = Users.GetAllUsers();

GridView1.DataBind();

}

}

**Audio.aspx.cs**public partial class Audio : System.Web.UI.Page

{

public int currentpage;

PagedDataSource pg;

DataSet ds;

//fires on page load

protected void Page\_Load(object sender, EventArgs e)

{

if (Request.QueryString["id"]!=null) // check cuz shuoldn't be null

{

if (Session["ID"] == null)

{

HideCommentAdding();

}

int audio\_id = int.Parse(Request.QueryString["id"]);

if (!IsPostBack)

{

this.currentpage = 0;

Session["CurrentPageComments"] = 0;

RefreshSong(audio\_id);

ShowComments(audio\_id);

}

else

{

this.currentpage = (int)(Session["CurrentPageComments"]);

}

int pageInd = this.currentpage + 1; //Page number

LabelCurrentPage.Text = pageInd.ToString();

}

else

{

LabelError.Text = "No Id Specified";

}

}

//refreshes data list of song

protected void RefreshSong(int audio\_id)

{

if (Session[audio\_id.ToString()] == null) //adds listen if user didn't listen

{

AudioFiles.AddListen(audio\_id);

Session[audio\_id.ToString()] = true;

}

DataList1.DataSource = AudioFiles.GetFileById(audio\_id);

DataList1.DataBind();

}

//refreshes data list of comments

protected void ShowComments(int audio\_id)

{

pg = new PagedDataSource();

ds = Comments.GetCommentsByAudioId(audio\_id);

if (ds.Tables[0].Rows.Count > 0)

{

pg.AllowPaging = true;

pg.DataSource = ds.Tables[0].DefaultView;

pg.PageSize = 4;

pg.CurrentPageIndex = this.currentpage;

this.ImageButtonNextPage.Enabled = !pg.IsLastPage;

this.ImageButtonPrevPage.Enabled = !pg.IsFirstPage;

DataListComments.DataSource = pg;

DataListComments.DataBind();

}

}

//returns picture for like button

protected string DidUserLikePic()

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], (int)Eval("ID")))

{

return "~/Images/SongIcons/LikeButton2.PNG";

}

}

return "~/Images/SongIcons/LikeButton.PNG";

}

// likes/dislikes

public void ImageButtonLike\_Click(object sender, ImageClickEventArgs e)

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText))) //AlternateText contains audio id

{

Likes.Unlike((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton.PNG";

RefreshSong(int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

}

else

{

Likes.Like((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton2.PNG";

RefreshSong(int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

}

}

else

{

Session["CurrentlyListening"] = int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText);

Response.Redirect("Login.aspx?m=You need an account to like audios");

}

}

//downloads audio

public void ImageButtonDownload\_Click(object sender, ImageClickEventArgs e)

{

int id = int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText);

Byte[] file = AudioFiles.Download(id);

System.Web.HttpContext context = System.Web.HttpContext.Current;

context.Response.Clear();

context.Response.ClearHeaders();

context.Response.ClearContent();

context.Response.AppendHeader("content-length", file.Length.ToString());

context.Response.ContentType = "audio/mpeg";

context.Response.AppendHeader("content-disposition", "attachment; filename=" + AudioFiles.GetFilenameById(id));

context.Response.BinaryWrite(file);

context.ApplicationInstance.CompleteRequest();

}

//goes to prev page

public void ImageButtonPrevPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageComments"] = (int)Session["CurrentPageComments"] - 1;

this.currentpage = (int)Session["CurrentPageComments"];

ShowComments(int.Parse(Request.QueryString["id"]));

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

//goes to next page

public void ImageButtonNextPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageComments"] = (int)Session["CurrentPageComments"] + 1;

this.currentpage = (int)Session["CurrentPageComments"];

ShowComments(int.Parse(Request.QueryString["id"]));

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

//submits a comment

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

{

string comment = TextBoxComment.Text.ToString();

int audio\_id = int.Parse(Request.QueryString["id"].ToString());

int user\_id = int.Parse(Session["ID"].ToString());

DateTime date = DateTime.Now;

if (Comments.DoesCommentExist(user\_id, audio\_id))

{

Comments.UpdateComment(user\_id, audio\_id, date, comment);

}

else

{

Comments.InsertComment(user\_id, audio\_id, date, comment);

}

ShowComments(int.Parse(Request.QueryString["id"].ToString()));

}

//hides option of commenting

protected void HideCommentAdding()

{

LabelCommentAdd.Text = "You need to signin in order to add a comment!";

TextBoxComment.Visible = false;

ButtonComment.Visible = false;

}

}

**Explore.aspx.cs**public partial class Explore : System.Web.UI.Page

{

public int currentpage;

PagedDataSource pg;

DataSet ds;

//Fires when page loads

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

this.currentpage = 0;

Session["CurrentPage"] = 0;

PopulateDataList(); //Binding pg to datalist

}

else

{

this.currentpage = (int)(Session["CurrentPage"]);

string idPlay = Request["\_\_EVENTARGUMENT"];

string target = Request["\_\_EVENTTARGET"];

if (target == "audio")

{

Response.Redirect("Audio.aspx?id="+idPlay);

}

}

int pageInd = this.currentpage + 1; //Page number

LabelCurrentPage.Text = pageInd.ToString();

}

//Refreshes datalist

protected void PopulateDataList()

{

pg = new PagedDataSource();

ds=AudioFiles.GetAllFiles();

if (ds.Tables[0].Rows.Count > 0)

{

pg.AllowPaging = true;

pg.DataSource = ds.Tables[0].DefaultView;

pg.PageSize = 5;

pg.CurrentPageIndex = this.currentpage;

this.ImageButtonNextPage.Enabled = !pg.IsLastPage;

this.ImageButtonPrevPage.Enabled = !pg.IsFirstPage;

DataList1.DataSource = pg;

DataList1.DataBind();

}

}

//returns image for like button

protected string DidUserLikePic()

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], (int)Eval("ID")))

{

return "~/Images/SongIcons/LikeButton2.PNG";

}

}

return "~/Images/SongIcons/LikeButton.PNG";

}

//likes/dislikes

public void ImageButtonLike\_Click(object sender, ImageClickEventArgs e)

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText))) //AlternateText contains audio id

{

Likes.Unlike((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton.PNG";

PopulateDataList();

}

else

{

Likes.Like((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton2.PNG";

PopulateDataList();

}

}

else

{

Response.Redirect("Login.aspx?m=You need an account to like audios");

}

}

//downloads file

public void ImageButtonDownload\_Click(object sender, ImageClickEventArgs e)

{

int id = int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText);

Byte[] file = AudioFiles.Download(id);

System.Web.HttpContext context = System.Web.HttpContext.Current;

context.Response.Clear();

context.Response.ClearHeaders();

context.Response.ClearContent();

context.Response.AppendHeader("content-length", file.Length.ToString());

context.Response.ContentType = "audio/mpeg";

context.Response.AppendHeader("content-disposition", "attachment; filename=" + AudioFiles.GetFilenameById(id));

context.Response.BinaryWrite(file);

context.ApplicationInstance.CompleteRequest();

}

//goes to prev page

public void ImageButtonPrevPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPage"] = (int)Session["CurrentPage"] - 1;

this.currentpage = (int)Session["CurrentPage"];

PopulateDataList();

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

//goes to next page

public void ImageButtonNextPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPage"] = (int)Session["CurrentPage"] + 1;

this.currentpage = (int)Session["CurrentPage"];

PopulateDataList();

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString(); } }

**Login.aspx.cs**public partial class Login : System.Web.UI.Page

{

//fires when page loads

protected void Page\_Load(object sender, EventArgs e)

{

LabelError.Text = Request.QueryString["m"];

}

//fires when submit button is clicked

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

{

if (Users.LoginSuccessful(TextBoxUsername.Text, TextBoxPassword.Text))

{

Session["Username"] = TextBoxUsername.Text;

Session["ID"] = Users.GetIdByUsername(TextBoxUsername.Text);

Session["Admin"] = Users.IsAdmin(int.Parse(Session["ID"].ToString()));

Application["connected"] = int.Parse(Application["connected"].ToString()) + 1;

if (Session["CurrentlyListening"] == null)

{

Response.Redirect("Home.aspx");

}

else

{

Response.Redirect("Audio.aspx?id=" + Session["CurrentlyListening"]);

}

}

else

LabelError.Text = "Incorrect Login";

}

}

**Logout.aspx.cs**public partial class Logout : System.Web.UI.Page

{

//Fires on page load

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["ID"] != null)

{

Session["Username"] = null;

Session["ID"] = null;

Session["Admin"] = null;

Application["connected"] = int.Parse(Application["connected"].ToString()) - 1;

}

Response.Redirect("Home.aspx");

}

}

**MasterPage.master.cs**public partial class MasterPage : System.Web.UI.MasterPage

{

public string welcome="";

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] == null) { welcome = "Welcome, Guest."; } else { welcome = "Welcome, " + Session["Username"].ToString() + "."; }

}

}

**MyLikes.aspx.cs**public partial class MyLikes : System.Web.UI.Page

{

public int currentpage;

PagedDataSource pg;

DataSet ds;

//fires when page loads

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["ID"] != null)

{

if (!IsPostBack)

{

this.currentpage = 0;

Session["CurrentPageLikes"] = 0;

PopulateDataList(); //Binding pg to datalist

}

else

{

this.currentpage = (int)(Session["CurrentPageLikes"]);

string idPlay = Request["\_\_EVENTARGUMENT"];

string target = Request["\_\_EVENTTARGET"];

if (target == "audio")

{

Response.Redirect("Audio.aspx?id=" + idPlay);

}

}

int pageInd = this.currentpage + 1; //Page number

LabelCurrentPage.Text = pageInd.ToString();

}

else

Response.Redirect("Home.aspx");

}

//refreshes datalist

protected void PopulateDataList()

{

pg = new PagedDataSource();

ds = Likes.GetLikesForUser(int.Parse(Session["id"].ToString()));

if (ds.Tables[0].Rows.Count > 0)

{

pg.AllowPaging = true;

pg.DataSource = ds.Tables[0].DefaultView;

pg.PageSize = 5;

pg.CurrentPageIndex = this.currentpage;

this.ImageButtonNextPage.Enabled = !pg.IsLastPage;

this.ImageButtonPrevPage.Enabled = !pg.IsFirstPage;

DataList1.DataSource = pg;

DataList1.DataBind();

}

else

LabelLikes.Text = "You didn't like any audios :(";

}

//returns image for like button

protected string DidUserLikePic()

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], (int)Eval("ID")))

{

return "~/Images/SongIcons/LikeButton2.PNG";

}

}

return "~/Images/SongIcons/LikeButton.PNG";

}

//likes/dislikes

public void ImageButtonLike\_Click(object sender, ImageClickEventArgs e)

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText))) //AlternateText contains audio id

{

Likes.Unlike((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton.PNG";

PopulateDataList();

}

else

{

Likes.Like((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton2.PNG";

PopulateDataList();

}

}

else

{

Response.Redirect("Login.aspx?m=You need an account to like audios");

}

}

//downloads file

public void ImageButtonDownload\_Click(object sender, ImageClickEventArgs e)

{

string webserviceurl = "";

Response.Redirect(webserviceurl + "?id=" + ((System.Web.UI.WebControls.Image)(sender)).AlternateText); //redirects to webservice to download file

}

//goes to prev page

public void ImageButtonPrevPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageLikes"] = (int)Session["CurrentPageLikes"] - 1;

this.currentpage = (int)Session["CurrentPageLikes"];

PopulateDataList();

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

//goes to next page

public void ImageButtonNextPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageLikes"] = (int)Session["CurrentPageLikes"] + 1;

this.currentpage = (int)Session["CurrentPageLikes"];

PopulateDataList();

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

}

**Search.aspx.cs**public partial class Search : System.Web.UI.Page

{

public int currentpage;

PagedDataSource pg;

DataSet ds;

string searchstring;

//fires when page loads

protected void Page\_Load(object sender, EventArgs e)

{

ClientScript.GetPostBackEventReference(this, string.Empty);

if (Request.QueryString["q"] != null) // check cuz shuoldn't be null

{

searchstring = Request.QueryString["q"];

LabelUp.Text = "Search results for: " + searchstring;

if (!IsPostBack)

{

this.currentpage = 0;

Session["CurrentPageSearch"] = 0;

PopulateDataList(); //Binding pg to datalist

}

else

{

this.currentpage = (int)(Session["CurrentPageSearch"]);

string idPlay = Request["\_\_EVENTARGUMENT"];

string target = Request["\_\_EVENTTARGET"];

if (target == "audio")

{

Response.Redirect("Audio.aspx?id=" + idPlay);

}

}

int pageInd = this.currentpage + 1; //Page number

LabelCurrentPage.Text = pageInd.ToString();

}

else

{

LabelUp.Text = "No search string specified";

}

}

//refreshes data list

protected void PopulateDataList()

{

pg = new PagedDataSource();

ds = AudioFiles.Search(searchstring);

if (ds.Tables[0].Rows.Count > 0)

{

pg.AllowPaging = true;

pg.DataSource = ds.Tables[0].DefaultView;

pg.PageSize = 5;

pg.CurrentPageIndex = this.currentpage;

this.ImageButtonNextPage.Enabled = !pg.IsLastPage;

this.ImageButtonPrevPage.Enabled = !pg.IsFirstPage;

DataList1.DataSource = pg;

DataList1.DataBind();

}

}

//returns pic for like button

protected string DidUserLikePic()

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], (int)Eval("ID")))

{

return "~/Images/SongIcons/LikeButton2.PNG";

}

}

return "~/Images/SongIcons/LikeButton.PNG";

}

//likes/dislikes

public void ImageButtonLike\_Click(object sender, ImageClickEventArgs e)

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText))) //AlternateText contains audio id

{

Likes.Unlike((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton.PNG";

PopulateDataList();

}

else

{

Likes.Like((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton2.PNG";

PopulateDataList();

}

}

else

{

Response.Redirect("Login.aspx?m=You need an account to like audios");

}

}

//downloads file

public void ImageButtonDownload\_Click(object sender, ImageClickEventArgs e)

{

int id = int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText);

Byte[] file = AudioFiles.Download(id);

System.Web.HttpContext context = System.Web.HttpContext.Current;

context.Response.Clear();

context.Response.ClearHeaders();

context.Response.ClearContent();

context.Response.AppendHeader("content-length", file.Length.ToString());

context.Response.ContentType = "audio/mpeg";

context.Response.AppendHeader("content-disposition", "attachment; filename=" + AudioFiles.GetFilenameById(id));

context.Response.BinaryWrite(file);

context.ApplicationInstance.CompleteRequest();

}

//goes to prev page

public void ImageButtonPrevPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageSearch"] = (int)Session["CurrentPageSearch"] - 1;

this.currentpage = (int)Session["CurrentPageSearch"];

PopulateDataList();

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

//goes to next page

public void ImageButtonNextPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageSearch"] = (int)Session["CurrentPageSearch"] + 1;

this.currentpage = (int)Session["CurrentPageSearch"];

PopulateDataList();

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

}

**Signup.aspx.cs**public partial class Signup : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

//fires when clicking submit button

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

{

if (Page.IsValid)

{

string error = Users.AddUser(TextBoxNickName.Text, TextBoxMail.Text, TextBoxPassword.Text, TextBoxUsername.Text, TextBoxRealName.Text);

if (error != "1")

LabelExistError.Text = error;

else

{

Response.Redirect("Login.aspx?m=Signup Successful!");

}

}

}

}

**Upload.aspx.cs**public partial class Upload : System.Web.UI.Page

{

//redirects home if not logged in

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] == null)

Response.Redirect("Home.aspx");

}

//fires when clicking button1, uploads file

protected void Button1\_Click(object sender, EventArgs e)

{

if (FileUpload1.HasFile)

Label1.Text = AudioFiles.UploadFile(FileUpload1, TextBoxAudioName.Text, TextBoxDescription.Text, Session["ID"].ToString());

else

Label1.Text = "You have not specified a file."; } }

**User.aspx.cs**public partial class User : System.Web.UI.Page

{

public int currentpage;

PagedDataSource pg;

DataSet ds;

int user\_id;

//fires when page loads

protected void Page\_Load(object sender, EventArgs e)

{

ClientScript.GetPostBackEventReference(this, string.Empty);

if (Request.QueryString["id"] != null) // check cuz shuoldn't be null

{

user\_id = int.Parse(Request.QueryString["id"]);

LabelUp.Text = Users.GetUserNickById(user\_id)+"'s Sound Page";

if (!IsPostBack)

{

this.currentpage = 0;

Session["CurrentPageUser"] = 0;

PopulateDataList(user\_id);

}

else

{

this.currentpage = (int)(Session["CurrentPageUser"]);

string idPlay = Request["\_\_EVENTARGUMENT"];

string target = Request["\_\_EVENTTARGET"];

if (target == "audio")

{

Response.Redirect("Audio.aspx?id=" + idPlay);

}

}

int pageInd = this.currentpage + 1; //Page number

LabelCurrentPage.Text = pageInd.ToString();

}

else

{

LabelUp.Text = "No Id Specified";

}

}

//refreshes data list

protected void PopulateDataList(int user\_id)

{

pg = new PagedDataSource();

ds = AudioFiles.GetAllFilesByUser(user\_id);

if (ds.Tables[0].Rows.Count>0)

{

pg.AllowPaging = true;

pg.DataSource = ds.Tables[0].DefaultView;

pg.PageSize = 5;

pg.CurrentPageIndex = this.currentpage;

this.ImageButtonNextPage.Enabled = !pg.IsLastPage;

this.ImageButtonPrevPage.Enabled = !pg.IsFirstPage;

DataList1.DataSource = pg;

DataList1.DataBind();

}

else

LabelNoAudio.Text = "This user didn't upload any files :(";

}

//returns image for like button

protected string DidUserLikePic()

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], (int)Eval("ID")))

{

return "~/Images/SongIcons/LikeButton2.PNG";

}

}

return "~/Images/SongIcons/LikeButton.PNG";

}

//likes/dislikes

public void ImageButtonLike\_Click(object sender, ImageClickEventArgs e)

{

if (Session["ID"] != null)

{

if (Likes.DidUserLikeAudio((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText))) //AlternateText contains audio id

{

Likes.Unlike((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton.PNG";

PopulateDataList(user\_id);

}

else

{

Likes.Like((int)Session["ID"], int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText));

((System.Web.UI.WebControls.Image)(sender)).ImageUrl = "~/Images/SongIcons/LikeButton2.PNG";

PopulateDataList(user\_id);

}

}

else

{

Response.Redirect("Login.aspx?m=You need an account to like audios");

}

}

//downloads file

public void ImageButtonDownload\_Click(object sender, ImageClickEventArgs e)

{

int id = int.Parse(((System.Web.UI.WebControls.Image)(sender)).AlternateText);

Byte[] file = AudioFiles.Download(id);

System.Web.HttpContext context = System.Web.HttpContext.Current;

context.Response.Clear();

context.Response.ClearHeaders();

context.Response.ClearContent();

context.Response.AppendHeader("content-length", file.Length.ToString());

context.Response.ContentType = "audio/mpeg";

context.Response.AppendHeader("content-disposition", "attachment; filename=" + AudioFiles.GetFilenameById(id));

context.Response.BinaryWrite(file);

context.ApplicationInstance.CompleteRequest();

}

//goes to prev page

public void ImageButtonPrevPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageUser"] = (int)Session["CurrentPageUser"] - 1;

this.currentpage = (int)Session["CurrentPageUser"];

PopulateDataList(user\_id);

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

//goes to next page

public void ImageButtonNextPage\_Click(object sender, ImageClickEventArgs e)

{

Session["CurrentPageUser"] = (int)Session["CurrentPageUser"] + 1;

this.currentpage = (int)Session["CurrentPageUser"];

PopulateDataList(user\_id);

int pageInd = this.currentpage + 1;

this.LabelCurrentPage.Text = pageInd.ToString();

}

}

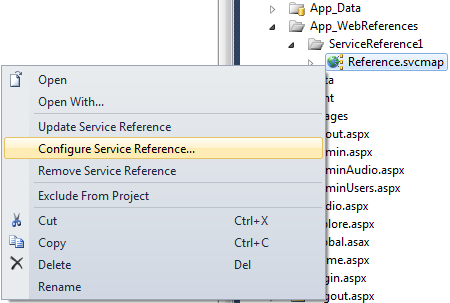
# **ReadMe**

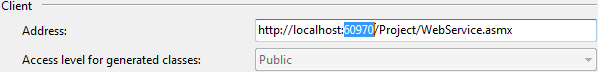
WebService

על מנת לוודא שהווב סרוויס יעבוד עם הפורט הנכון, יש להריץ את האתר פעם אחת, ולראות בשורת הכתובות את הפורט בו האתר משתמש.



במקרה זה הפורט הוא 5499.  
לאחר מכן, יש לגשת לSolution Explorer, למצוא את הרפרנס לווב סרוויס, להקליק קליק ימיני בעכבר, וללחוץ על Configure Service Reference.



שם, יש לשנות את הפורט בהתאם לפורט שמצאנו קודם.  


במקרה שלא ניתן להוריד קבצים, ומופיעה שגיאה לגבי MaxRequestLength, יש לבצע את הצעדים הבאים:  
1. לפתוח את 

2.לוודא שהשורה המגדירה את הווב סרוויס נראית כך:



Users Login  
כדי להתחבר בתור משתמש רגיל: כדי להתחבר בתור אדמין:

Username: dan2  
Password: dan

Username: dan  
Password: dan