משחק חלליות / 

**עבודת גמר**

**בתכנון ותכנות מערכות**

**במסלול שירותי רשת אינטרנט**

**שנת הלימודים**

|  |
| --- |
| שם התלמיד :עדי נזרי |
| בית ספר : מקיף י"ב המעיין |
| ת.ז :212252126 |
| שם המורה :חנה לוי |

## הקדשה/תודה:

## ראשית כל, אני רוצה להקדיש את הפרויקט שלי למורתי חנה,ששלב אחרי שלב, לימדה אותי בצורה מובנת ומסודרת את כל החומר.

## מעבר לפרויקט שהגיע לידי סיום בהצלחה, למדתי והבנתי את כל נושא אתרי האינטרנט ואוכל להשתמש בו בעתיד.

## תוכן עניינים

[הקדשה/תודה: 2](#_Toc120266094)

[תוכן עניינים 3](#_Toc120266095)

[מבוא 4](#_Toc120266096)

[מטרות האתר 4](#_Toc120266097)

[קהל יעד 4](#_Toc120266098)

[תאור האתר 4](#_Toc120266099)

[מבנה בסיס הנתונים 5-8](#_Toc120266100)

[תמונת הטבלאות מתוך מסד הנתונים 5-8](#_Toc120266101)

[תמונת קשרי הגומלין מתוך מסד הנתונים ( DSD) 9](#_Toc120266102)

[מבנה התיקיות 10](#_Toc120266103)

[מחלקות 11-34](#_Toc120266103)

[שרותי הרשת באתר 35-39](#_Toc120266104)

[מבנה בסיס הנתונים של שירותי הרשת 40-41](#_Toc120266107)

[תמונת הטבלאות מתוך מסד הנתונים 40-41](#_Toc120266108)

[מפת האתר 42](#_Toc120266110)

[מדריך האפליקציה למשתמש 43-46](#_Toc120266111)

[מדריך האפליקציה למנהל 47-48](#_Toc120266112)

[נספחים 49](#_Toc120266114)

[ReadMe 50](#_Toc120266115)

## מבוא

### מטרות האתר:

האתר מבוסס על משחק חלליות ומטרת המשחק היא לנצח את הבוס בזמן הקצר ביותר.

משחק ניתן לעלות רמות ולצבור נקודות שאיתן אפשר לקנות חלליות טובות יותר אשר יעזרו לשחקנים לנצח מהר יותר, בנוסף בעזרת הנקודות ניתן לשפר את מהירות החלליות ואת כוחן.

מטרת האתר היא כמטרת כל משחק. להנות.

### קהל יעד

קהל היעד הוא ילדים או אנשים מבוגרים אשר חובבים משחקים בסיגנון משחקי אטארי של פעם.

### תאור האתר

ראשית כל לפני שנכנסים לאתר יש להירשם ולהתחבר לאתר. לאחר ההתחברות יש למשתמש גישה לאתר שבוא הוא יכול לשחק במשחק ולצבור נקודות ולשבור שיאים. משתמש אשר ניצח בזמן קצר יופיע בטבלת עשרת הטובים החשופה לכולם.

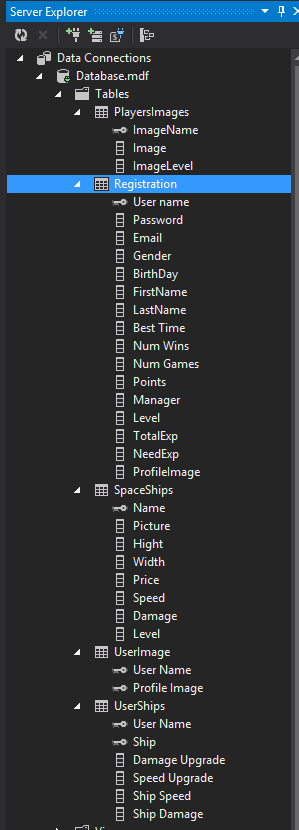
משתמש אשר רוצה לנצח להופיע מהר בטבלה זאת יכול לרכוש נקודות אשר יעזרו לו לשפר את החללית ובכך יוכל לנצח בזמן טוב יותר ולהתברג יותר בקלות בטבלה.

בנוסף ככל שמתקדמים ברמות צוברים תמונות פרופיל המעידות על רמת המשתמש.

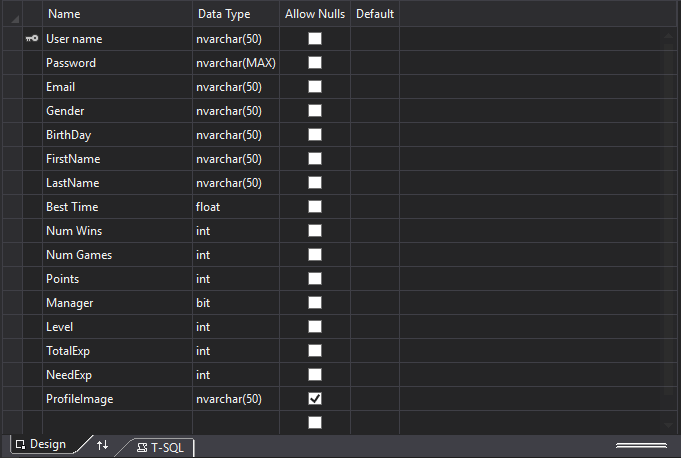
המשתמש יכול לבחור את תמונת הפרופיל שלו

## מבנה בסיס הנתונים

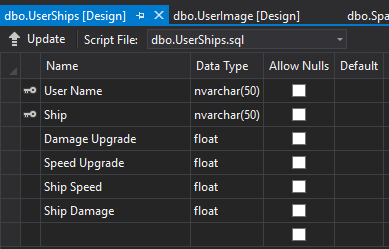
### תמונת/תמונות הטבלאות מתוך מסד הנתונים



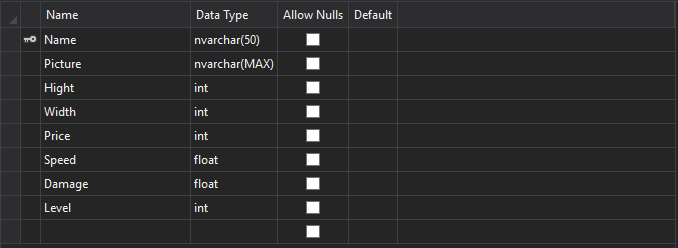
טבלת נתוני המשתמש המשתמש [Registration]



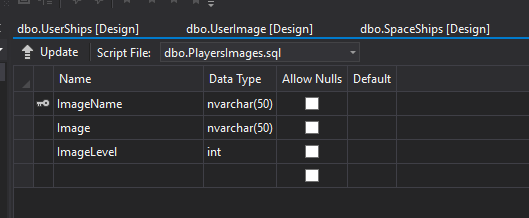
חלליות המשתמש[UserShips]



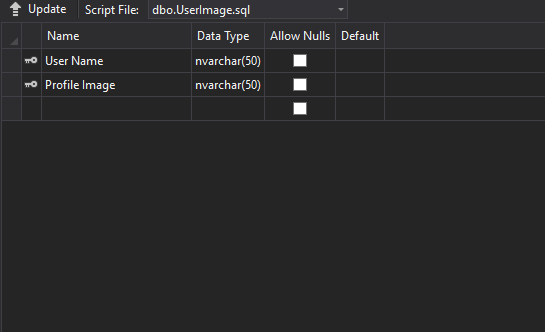
טבלת החלליות[SpaceShips]



טבלת תמונות הפרופיל[PlayersImages]



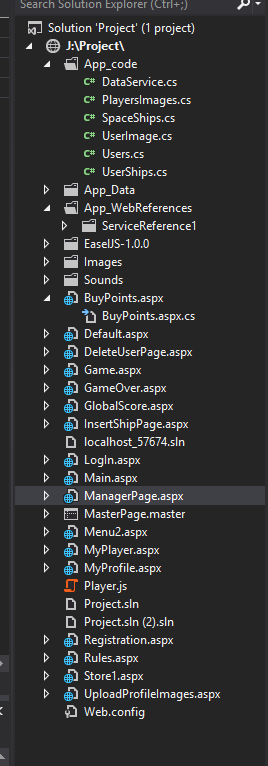
טבלת התמונות של המשתמש[UserImage]



### 

### תמונת/תמונות קשרי הגומלין מתוך מסד הנתונים ( DSD)

## D:\downloads\MyProject (1).jpegמבנה התיקיות

המחלקות

**המחלקה Data Service**

public class DataService

{

static private string path;

public DataService()

{

//

// TODO: Add constructor logic here

//

}

static public string Path

{

set

{

path = value;

}

get

{

return path;

}

}

static private string ConnectionString

{

get

{

return @"Data Source=(LocalDB)\v11.0;AttachDbFilename='"+DataService.path+"';Integrated Security=True";

}

}

//פעולה להוספה עדכון מחיקה

static public object ExecuteScalar(string strSql1)

{

object obj = null;

SqlConnection connection = new SqlConnection(DataService.ConnectionString);

SqlCommand command = new SqlCommand(strSql1, connection);

try

{

connection.Open();

obj = command.ExecuteScalar();

}

catch (Exception ex)

{

return ex;

}

finally

{

connection.Close();

}

return obj;

}

// פעולה להוספה עדכון ומחיקה

static public int ExectureNonQuery(string strSql1)

{

int rowsAffected;

SqlConnection connection = new SqlConnection(DataService.ConnectionString);

SqlCommand command = new SqlCommand(strSql1, connection);

try

{

connection.Open();

rowsAffected = command.ExecuteNonQuery();

}

catch (Exception ex)

{

return -1;

}

finally

{

connection.Close();

}

return rowsAffected;

}

// פעולה להצגת יותר מנתון אחד בצורה מחוברת

static public object DataReader(string strSql)

{

SqlConnection con = new SqlConnection(DataService.ConnectionString);

SqlCommand mycmd = new SqlCommand(strSql, con);

SqlDataReader dr = null;

string output = "";

try

{

con.Open();

dr = mycmd.ExecuteReader();

while (dr.Read())

{// יש לבנות אותה לפי הנתונים בטבלה

output += dr["[User name]"].ToString();

output += "<br>" + dr["Password"].ToString();

output += "<br>" + dr["Email"].ToString();

}

}

catch (Exception es)

{

return es;

}

finally

{

dr.Close();

con.Close();

}

return output;

}

// פעולה להצגת יותר מנתון אחד בצורה מנותקת

public static DataSet GetDataSet(string sql, string tblName)

{

DataSet ds = new DataSet();

SqlConnection con = new SqlConnection(DataService.ConnectionString);

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter ad = new SqlDataAdapter(cmd);

try

{

ad.Fill(ds, tblName);

}

catch (Exception ex)

{

return null;

}

finally

{

}

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

return null;

return ds;

}

}

**המחלקה users**

public class Users

{

private string userName;// שם משתמש

private string password;//סיסמא

private string email;//אימייל

private string gender;//מין

private string birthDay;//יום הולדת

private string firstName;//שם פרטי

private string lastName;//שם משפחה

private double bestTime;//הזמן הכי טוב של השחקן

private int numWins;//מספר נצחונות

private int numGames;//מספר משחקים

private int points;//מספר נקודות השחקן

private int manager;//האם מנהל

private int level;//רמת השחקן

private int totalExp;//כמות הexp

private int needExp;// כמות הexp הדרושה לעליית רמה

private string profileImg;

//פעולה בונה

public Users(string userName, string password, string email, string gender, string birthDay, string firstName, string lastName,int manager)

{

this.userName = userName;

this.password = password;

this.email = email;

this.gender = gender;

this.birthDay = birthDay;

this.firstName = firstName;

this.lastName = lastName;

this.bestTime = 0;

this.numWins = 0;

this.numGames = 0;

this.points = 0;

this.manager = manager;

this.level = 1;

this.totalExp = 0;

this.needExp = 1000;

this.profileImg = "beginner";

}

public string UserName

{

get

{

return this.userName;

}

set

{

this.userName = value;

}

}

public string Password

{

get

{

return this.password;

}

set

{

this.password = value;

}

}

public string Email

{

get

{

return this.email;

}

set

{

this.email = value;

}

}

public string Gender

{

get

{

return this.gender;

}

set

{

this.gender = value;

}

}

public string BirthDay

{

get

{

return this.birthDay;

}

set

{

this.birthDay = value;

}

}

public string FirstName

{

get

{

return this.firstName;

}

set

{

this.firstName = value;

}

}

public string LasttName

{

get

{

return this.lastName;

}

set

{

this.lastName = value;

}

}

public bool Insert()//מוסיפה משתמש לממסד הנתונים

{

string query1 = "insert into [Registration] values('" + this.userName + "','" + this.password + "','" + this.email + "','" + this.gender + "','" + this.birthDay + "','" + this.firstName + "','" + this.lastName + "','" + "0" + "','" + 0 + "','" + 0 + "','" + 0 + "','" + this.manager + "','" + this.level + "','" + this.totalExp + "','" + this.needExp + "','"+this.profileImg+"')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}

//פעולת מחיקת משתמש

public static bool Delete(string userName)

{

string query = "Delete From[Registration] where([User name]= '" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה המעדכנת את פרטי המשתמש

public bool Update(string userName, int password, string email, string gender, string birthDay, string firstName, string lastName)

{

string query = "Update [Registration] set [User name]='" + userName + "', [Password]='" + password + "',[Email]='" + email + "',[Gender]='" + gender + "',[BirthDay]='" + birthDay + "',[FirstName]='" + firstName + "',[LastName]='" + lastName + "'where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה המעדכנת את כמות משחקי השחקן

public static bool UpdateNumGame(string userName)

{

string query = "Update [Registration] set [Num Games]=( [Num games]+'" + 1 + "') where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה הבודקת האם המשתמש הוא מנהל

public static bool IsManager(string userName)

{

string query = "select [User name] from [Registration] where ([User name] ='" + userName + "' AND [Manager]='"+true+"')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

else

{

return false;

}

}

//פעולה המעדכנת את נקודות השחקן

public static bool UpdatePoints(int points ,string userName)

{

string query = "Update [Registration] set [Points]=('" + points + "'+[Points]) where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה המעדכנת את כמות הexp הנדרשת כדי לעלות רמה

public static bool UpdateNeedExp(string userName)

{

string query = "Update [Registration] set [NeedExp]=(([Level]+1)\*1000)+[TotalExp] where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

public static bool UpdateLevel(int levels,string userName)

{

string query = "Update [Registration] set [Level]='" + levels + "' where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

public static bool UpdateTotalExp(int exp, string userName)

{

string query = "Update [Registration] set [TotalExp]='" + exp + "' where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה המעדכנת את נתוני השחקן לאחר משחק

public static bool UpdateAfterGame(int points, int win, string userName)

{

string query = "Update [Registration] set [Points]=('" + points + "'+[Points]) ,[Num Wins]=([Num Wins]+'" + win + "') where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה המעדכנת את זמן השחקן

public static bool UpdateBestTime(double best, string userName)

{

string query = "Update [Registration] set [Best Time]='" + best + "' where([User name]='" + userName + "' AND ([Best Time]>'" + best + "' OR [Best Time]='" + 0 + "'))";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

public static bool UpdateProfileImage(string userName,string img)

{

string query = "Update [Registration] set [ProfileImage]='" +img + "' where([User name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

//פעולה המחזירה את מספר הנצחונות

public static int GetWins(string userName)

{

string query = "select [Num Wins] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

//פעולה המחזירה את מספר המשחקים

public static int GetGames(string userName)

{

string query = "select [Num Games] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

//פעולה המחזירה את כמות הנקדות של השחקן

public static int GetPoints(string userName)

{

string query = "select [Points] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

//פעולה המחזירה את הזמן הטוב ביותר של השחקן

public static string GetBestTime(string userName)

{

string query = "select [Best Time] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return obj.ToString();

}

return "0";

}

//פעולה המחזירה את כמות הexp של השחקן

public static int GetTotalExp(string userName)

{

string query = "select [TotalExp] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

//פעולה המחזירה את שם תמונת הפרופיל של השחקן

public static string GetProfileImage(string userName)

{

string query = "select [ProfileImage] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return obj.ToString();

}

return "";

}

//פעולה המחזירה את כמות הexp הנדרשת כדי לעלות רמה

public static int GetNeededExp(string userName)

{

string query = "select [NeedExp] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

//פעולה המחזירה את כמות הexp של השחקן

public static int GetHaveExp(string userName)

{

string query = "select ([TotalExp]-([Level]-1)\*1000) from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

//פעולה המחזירה את רמת השחקן

public static int GetLevel(string userName)

{

string query = "select [Level] from [Registration] where ([User name]='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return int.Parse(obj.ToString());

}

return 0;

}

// פעולה הבודקת האם ההמשתמש קיים לפי שם משתמש

public static bool Existed(string userName)

{

string query = "select [User name] from [Registration] where ([User name] ='" + userName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

else

{

return false;

}

}

// פעולה הבודקת האם ההמשתמש קיים לפי שם משתמש וסיסמא

public static bool Existed(string userName, string pass)

{

string query = "select [User name] from [Registration] where ([User name] ='" + userName + "' and [Password]='" + pass + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

return false;

}

//פעולה המחזריה את נתוני כל המשתמשים

public static DataSet GetUsers()

{

string query = "select\* from [Registration]";

DataSet ds = DataService.GetDataSet(query, "Users");

if (ds.Tables["Users"].Rows.Count == 0)

{

return null;

}

return ds;

}

// פעולה המחזירה את נתוני המשתמשים לפי סדר זמן הכי טוב (הכי קצר). א

public static DataSet GetUsersByTime()

{

string query = "select\* from [Registration] where ([Best Time]> 0 ) order by([Best Time]) ";

DataSet ds = DataService.GetDataSet(query, "Users");

if (ds.Tables["Users"].Rows.Count == 0)

{

return null;

}

return ds;

}

//פעולה המחזירה דטה סט של המשתמש

public static DataSet GetUserData(string userName)

{

string query = "Select \* from[Registration],[PlayersImages] where ([User name] ='" + userName + "' AND [ProfileImage]=[ImageName])";

DataSet ds = DataService.GetDataSet(query, "Users");

if (ds.Tables["Users"].Rows.Count == 0)

{

return null;

}

return ds;

}

//מחזירה משתמש לפי השם משתמש

public static Users GetUserByUserName(string userName)

{

string query = "Select \* from[Registration] where ([User name] ='" + userName + "')";

DataSet ds = DataService.GetDataSet(query, "UserByUserName");

if (ds.Tables["UserByUserName"].Rows.Count == 0)

{

return null;

}

else

{

string firstName = ds.Tables["UserByUserName"].Rows[0]["FirstName"].ToString();

string lastName = ds.Tables["UserByUserName"].Rows[0]["LastName"].ToString();

string email = ds.Tables["UserByUserName"].Rows[0]["Email"].ToString();

string gender = ds.Tables["UserByUserName"].Rows[0]["Gender"].ToString();

string password = (ds.Tables["UserByUserName"].Rows[0]["Password"].ToString());

string birthDay = ds.Tables["UserByUserName"].Rows[0]["BirthDay"].ToString();

int numWins = int.Parse(ds.Tables["UserByUserName"].Rows[0]["Num Wins"].ToString());

int numGames = int.Parse(ds.Tables["UserByUserName"].Rows[0]["Num Games"].ToString());

double bestTime = double.Parse(ds.Tables["UserByUserName"].Rows[0]["Best Time"].ToString());

int manager = int.Parse(ds.Tables["UserByUserName"].Rows[0]["Manager"].ToString());

Users u = new Users(userName, password, email, gender, birthDay, firstName, lastName,manager);

return u;

}

}

}

## 

## המחלקה PlayersImagers

public class PlayersImages

{

private string imageName;//שם התמונה

private string image;//כתובת התמונה

private int level;//רמה הנדרשת כדי להקבל את התמונה

public PlayersImages(string name,string img,int level)

{

this.imageName = name;

this.image = img;

this.level = level;

}//פעולה הבונה עצם מסוג playersImages

public bool Insert()

{

string query = "insert into [PlayersImages] values('" + this.imageName + "','" + this.image + "','" + this.level + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}//מכינה את העצם לטבלת PlayersImages

public static bool DeleteByName(string name)

{

string query = "delete \* from [PlayersImages] where(ImageName = '" + name + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}//פעולה המקבלת שם של תמונה ומוחקת אותה מהטבלה PlayersImages

public static DataSet GetAllImages()

{

string query = "select \* from [PlayersImages]";

DataSet ds = DataService.GetDataSet(query, "PlayersImages");

if (ds.Tables["PlayersImages"].Rows.Count == 0)

return null;

return ds;

}//פעולה המחזירה דטה סט של המידע של כל התמונות מהטבלה

public static DataSet GetAllImagesExceptBeginner()

{

string query = "select \* from [PlayersImages] where(imageName!=beginner)";

DataSet ds = DataService.GetDataSet(query, "PlayersImages");

if (ds.Tables["PlayersImages"].Rows.Count == 0)

return null;

return ds;

}// פעולה המחזירה דטה סט של המידע של כל התמונות מהטבלה חוץ מהתמונה ששמה beginner

public static DataSet GetImagesByLevel(int level)

{

string query = "select \* from [PlayersImages] where([ImageLevel] <= '"+ level +"' AND [ImageName]!= 'beginner')";

DataSet ds = DataService.GetDataSet(query, "PlayersImages");

if (ds.Tables["PlayersImages"].Rows.Count == 0)

return null;

return ds;

}//פעולה שמקבלת רמה ומחזירה דטה סט של כל התמונות שמתחת או שוות לרמה שקיבלה

public static bool Exist(string name)

{

string query = "select ImageName from [PlayersImages] where([ImageName]='" + name + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return true;

return false;

}//פעולה הבודקת אם התמונה קיימת בטבלה

}

**המחלקה SpaceShips**

public class SpaceShips

{

private string name;// The ship name

private string picture=@"~\Images\";// the ship's picture img

private int high;//the ship's hight

private int width;//the ship's width

private int price;//the ship's price

private float speed;// the ship's speed

private float damage;// the ship's damage

private int level;// the ship's level

public SpaceShips(string name, string pic, int high, int width,int price,float speed,float damage,int level)

{

this.name = name;

this.picture = pic;

this.high = high;

this.width=width;

this.price = price;

this.speed = speed;

this.damage = damage;

this.level = level;

}// constructor

public bool Insert() // function that insert the object to the table

{

string query1 = "insert into [SpaceShips] values('" + this.name + "','" + this.picture + "','" + this.high + "','" + this.width + "','" + this.price + "','" + this.speed + "','" + this.damage + "','"+this.level+"')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}

public static bool Delete(string Name)

{

string query = "Delete From[SpaceShips] where([Name]= '" + Name + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

} // function that get ship's name and deleted the ship from the table

public static bool Existed(string Name)

{

string query = "select [Name] from [SpaceShips] where ([Name] ='" + Name + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

else

{

return false;

}

} //function that return if ship existed in the database

public static int GetHight(string name)

{

string query = "select [Hight] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return int.Parse(obj.ToString());

return 0;

}// function that return the ship's hight

public static int GetLevel(string name)

{

string query = "select [Level] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return int.Parse(obj.ToString());

return 0;

}// function that return the ship's level

public static float GetSpeed(string name)

{

string query = "select [Speed] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return float.Parse(obj.ToString());

return 0;

}// function that return the ship's speed

public static float GetDamage(string name)

{

string query = "select [Damage] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return float.Parse(obj.ToString());

return 0;

}//function that return the ship's damage

public static int GetWidth(string name)

{

string query = "select [Width] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return int.Parse(obj.ToString());

return 0;

}// function that return the ship's width

public static string GetPicture(string name)

{

string query = "select [Picture] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return obj.ToString();

return null;

}//function that return the ship's picture

public static int GetPrice(string name)

{

string query = "select [Price] from [SpaceShips] where([Name]= '" + name + "') ";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

return int.Parse(obj.ToString());

return 0;

} //function that return the ship's price

public static bool UpdatePicture(string name, string picture)

{

string pic = @"~\Images\" + picture;

string query = "Update [SpaceShips] set [Picture]='" + pic + "' where([Name]='" + name + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

} //function that update the ship's picture

public static bool UpdateAll(string name, string pic, int high, int width, int price, float speed, float damage,int level)

{

string query = "Update [SpaceShips] set [Picture]='" + pic + "',[Hight]='" + high + "',[Width]='"

+ width + "',[Price]='" + price + "',[Speed]='" + speed + "',[Damage]='" + damage + "',[Level]='"+level+"' where ([Name]='" + name + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}//function that update all the ship's valuse

public static DataSet GetSpaceShip()

{

string query = "select [Name],[Picture],[Hight],[Width],[Price],[Speed]\*100 AS [ShipSpeed],[Damage]\*100 AS [ShipDamage],[Level] from [SpaceShips]";

DataSet ds = DataService.GetDataSet(query, "SpaceShips");

if (ds.Tables["SpaceShips"].Rows.Count == 0)

{

return null;

}

return ds;

}//function that return data set of all the ships from the table

public static DataSet GetSpaceShipForStore()

{

string query = "select [Name],[Level],[Picture],[Hight],[Width],[Price],[Speed]\*100 AS [ShipSpeed],[Damage]\*100 AS [ShipDamage] from [SpaceShips] where [Name]!= 'Default Ship'";

DataSet ds = DataService.GetDataSet(query, "SpaceShips");

if (ds.Tables["SpaceShips"].Rows.Count == 0)

{

return null;

}

return ds;

}//function that return data set of all the ships except the defult ship from the table

public static DataSet GetSpaceShipByName(string ship)

{

string query = "select [Name],[Picture],[Hight],[Width],[Price]\*0.5 AS[SellPrice],[Speed]\*100 AS [ShipSpeed],[Damage]\*100 AS [ShipDamage] from [SpaceShips] where([Name]='"+ship+"')";

DataSet ds = DataService.GetDataSet(query, "SpaceShips");

if (ds.Tables["SpaceShips"].Rows.Count == 0)

{

return null;

}

return ds;

}//function that return data set of the requested ship from the table

}

**המחלקה USerShips**

public class UserShips

{

private string userName;//שם המשתמש

private string ShipName;//שם החללית

private float shipSpeed;//מהירות החללית

private float shipDamage;//כח החללית

private float speedUpgrade;//שידרוג מהירות

private float damageUpgrade;//שידרוג כח

public UserShips(string userName, string ShipName, float shipSpeed, float shipDamage, float speedUpgrade, float damageUpgrade)

{

this.userName = userName;

this.ShipName = ShipName;

this.shipSpeed = shipSpeed;

this.shipDamage = shipDamage;

this.speedUpgrade = speedUpgrade;

this.damageUpgrade = damageUpgrade;

}//פעולה בונה

public bool Insert()

{

string query1 = "insert into [UserShips] values('" + this.userName + "','" + this.ShipName + "','"+this.damageUpgrade+"','"+this.speedUpgrade+"','"+this.shipSpeed+"','"+this.shipDamage+"')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}//מכניסה את העצם לטבלה user Ships

public static bool CanBuyByLevel(string userName, string ShipName,int level)

{

string query = "select [Name] from [SpaceShips] where ([Level] <='" + level + "' And [Name] ='" + ShipName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

else

{

return false;

}

}//פעולה הבודקת אם השחקן יכול לקנות את החללית

public static bool Delete(string userName, string ShipName)

{

string query1 = "Delete from [UserShips] where ([User Name]='" + userName + "'And [Ship]='" + ShipName + "')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}//פעולה המוחקת חללית של שחקן מהטבלה

public static bool DeleteUser(string userName)

{

string query1 = "Delete from [UserShips] where ([User Name]='" + userName + "')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}//פעולה המוחקת את כל החלליות של השחקן מהטבלה

public static bool DeleteByShip(string ShipName)

{

string query1 = "Delete from [UserShips] where ([Ship]='" + ShipName + "')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}//פעולה המוחקת את החללית מכל המשתמשים

public static bool Existed(string userName, string ShipName)// פעולה הבודקת אם החללית קיימת עם השחקן בטבלה

{

string query = "select [User name] from [UserShips] where ([User name] ='" + userName + "' And [Ship] ='" + ShipName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

else

{

return false;

}

}

public static float GetShipSpeed(string userName, string ShipName)

{

string query = "select [Ship Speed] from [UserShips] where ([User name] ='" + userName + "' And [Ship] ='" + ShipName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return float.Parse(obj.ToString());

}

else

{

return 0;

}

}//פעולה המחזירה את מהירות החללית הנדרשת של השחקן

public static float GetShipDamage(string userName, string ShipName)

{

string query = "select [Ship Damage] from [UserShips] where ([User name] ='" + userName + "' And [Ship] ='" + ShipName + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return float.Parse(obj.ToString());

}

else

{

return 0;

}

}//פעולה המחזירה את הכח של החללית הנדרשת של השחקן

public static bool UpgradeSpeed(string ship, string userName)

{

string query = "Update [UserShips] set [Ship Speed]=ROUND([Ship Speed]\*1.03,2),[Speed Upgrade]=[Speed Upgrade]+1 where([Ship]='" + ship + "' AND [User Name]='" + userName + "' AND ([Damage Upgrade]+[Speed Upgrade]<13))";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

} // פעולה שמשדרגת את מהירות החללית הנדרשת של השחקן

public static bool UpgradeDamage(string ship, string userName)

{

string query = "Update [UserShips] set [Ship Damage]=ROUND([Ship Damage]\*1.03,2),[Damage Upgrade]=[Damage Upgrade]+1 where([Ship]='" + ship + "' AND [User Name]='" + userName + "' AND ([Damage Upgrade]+[Speed Upgrade]<13))";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}// פעולה שמשדרגת את כח החללית הנדרשת של השחקן

public static DataSet GetUserSpaceShip(string userName)

{

string query = "select [Ship] from [UserShips] where([User Name]='"+userName+"')";

DataSet ds = DataService.GetDataSet(query, "UserShips");

if (ds.Tables["UserShips"].Rows.Count == 0)

{

return null;

}

return ds;

} // פעולה המחזירה דטה סט של כל חלליות השחקן

public static DataSet GetSpaceShipByName(string ship ,string userName)

{

string query;

if (ship == "Default Ship")

{

query = "select [Ship],[Picture],CAST([Price]\*0.5 As INT) AS [SellPrice],[Ship Speed]\*100 AS [ShipSpeed],[Ship Damage]\*100 AS [ShipDamage],ROUND([Ship Damage]\*1.03\*100,0) AS[AfterDamageUpgrade],ROUND([Ship Speed]\*1.03\*100,0) AS[AfterSpeedUpgrade],15000\*0.5\*[Damage Upgrade] AS [DamageUpgradePrice],15000\*0.5\*[Speed Upgrade] As[SpeedUpgradePrice],14-[Speed Upgrade]-[Damage Upgrade] AS [UpgradesLeft] from [UserShips],[SpaceShips] where([Name]='" + ship + "' AND [User Name]='" + userName + "' AND [Ship]='" + ship + "')";

}

else

{

query = "select [Ship],[Picture],CAST([Price]\*0.5 As INT) AS[SellPrice],[Ship Speed]\*100 AS [ShipSpeed],[Ship Damage]\*100 AS [ShipDamage],ROUND([Ship Damage]\*1.03\*100,0) AS[AfterDamageUpgrade],ROUND([Ship Speed]\*1.03\*100,0) AS[AfterSpeedUpgrade],[Price]\*0.5\*[Damage Upgrade] AS [DamageUpgradePrice],[Price]\*0.5\*[Speed Upgrade] As[SpeedUpgradePrice],14-[Speed Upgrade]-[Damage Upgrade] AS [UpgradesLeft] from [UserShips],[SpaceShips] where([Name]='" + ship + "' AND [User Name]='" + userName + "' AND [Ship]='" + ship + "')";

}

DataSet ds = DataService.GetDataSet(query, "UserShips");

if (ds.Tables["UserShips"].Rows.Count == 0)

{

return null;

}

return ds;

}// פעולה המחזירה דטה סא של חחלי השחקן

}

**המחלקה UserImage**

public class UserImage

{

private string userName; //user name

private string profileImage;//The user optional Profile Image

public UserImage(string user,string img)

{

this.userName = user;

this.profileImage = img;

}

public bool Insert()// insert into the tabel UserImage the userName with his optional Profile Image

{

string query1 = "insert into [UserImage] values('" + this.userName + "','" + this.profileImage + "')";

int row = DataService.ExectureNonQuery(query1);

if (row > 0)

{

return true;

}

return false;

}

public static bool DeleteImageFromUser(string userName, string img)//Delete Image from the tabel in the user row

{

string query = "Delete From [UserImage] where([User Name]= '" + userName + "' AND [Profile Image]='" + img + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

public static bool DeleteImage(string img)//delete image from all users

{

string query = "Delete From [UserImage] where([Profile Image]='" + img + "')";

int row = DataService.ExectureNonQuery(query);

if (row > 0)

{

return true;

}

return false;

}

public static bool Existed(string userName, string img)//check if the user have the image

{

string query = "select [User Name] from [UserImage] where ([User name] ='" + userName + "' And [Profile Image] ='" + img + "')";

object obj = DataService.ExecuteScalar(query);

if (obj != null)

{

return true;

}

else

{

return false;

}

}

public static DataSet GetUserImages(string userName)// return all user's images

{

string query = "select [Profile Image],[Image],[ImageLevel] from [UserImage],[PlayersImages] where([User Name]='"+userName+"' And[ImageName]=[Profile Image])";

DataSet ds = DataService.GetDataSet(query, "UserImage");

if (ds.Tables["UserImage"].Rows.Count == 0)

{

return null;

}

return ds;

}

}

## שרותי הרשת באתר

### שירותי רשת נצרכים ממקור חיצוני

שימוש באותה המחלקה data service שתוארה למעלה.

<http://localhost:59391/webservice/CreditCard.asmx>

public class CreditAction

{

public CreditAction()

{

//

// TODO: Add constructor logic here

//

}

private string company;// שם חברת האשראי

private string cardNumber;//מספר כרטיס

private string validity;//תוקף כרטיס

private int cvv;// שלוש ספרות ביטחון

private float balance;// יתרה

private long id;//ת.ז של בעל הכרטיס

// פעולה בונה

public CreditAction(string company, string cardNumber, string validity, int cvv, float balance, long Id)

{

this.id = id;

this.company = company;

this.cardNumber = cardNumber;

this.validity = validity;

this.cvv = cvv;

this.balance = balance;

}

public string Company// שם חברת כרטיסהאשראי

{

get

{

return this.company;

}

set

{

this.company = value;

}

}

public string CardNumber// מספר כרטיס האשראי

{

get

{

return this.cardNumber;

}

set

{

this.cardNumber = value;

}

}

public string Validity// תוקף של כרטיס האשראי

{

get

{

return this.validity;

}

set

{

this.validity = value;

}

}

public int Cvv// מספר ביטחון של כרטיס האשראי

{

get

{

return this.cvv;

}

set

{

this.cvv = value;

}

}

public float Balance// יתרה

{

get

{

return this.balance;

}

set

{

this.balance = value;

}

}

public long Id// ת.ז של בעל כרטיס האשראי

{

get

{

return this.Id;

}

set

{

this.Id= value;

}

}

//פעולה המחזירה האם הכרטיס תקף

static public bool IsNotExpired(string cardNumber)

{

string cmd = "select [validity] From [CreditCards] where [cardNumber] = '" + cardNumber + "'";

object expire = DataService.ExecuteScalar(cmd);

string e = expire.ToString();

string year=""+e[3]+e[4]+e[5]+e[6];

string month=""+e[0]+e[1];

if (int.Parse(year) > int.Parse(DateTime.Now.Year.ToString()))

return true;

else if (int.Parse(year) == int.Parse(DateTime.Now.Year.ToString()))

{

if (int.Parse(month) > int.Parse(DateTime.Now.Month.ToString()))

{

return true;

}

else

return false;

}

return false;

}

//פעולה המחזירה האם הכרטיס קיים

static public bool Exist(string cardNumber)

{

string cmd = "select \* From [CreditCards] where [cardNumber] = '" + cardNumber + "'";

object obj = DataService.ExecuteScalar(cmd);

if (obj != null)

return true;

return false;

}

//פעולה המחזירה האם ניתן לשלם בכרטיס

static public bool Pay(string cardNumber, float cost)

{

bool CanPay = true;

string cmd1 = "select [balance] From [CreditCards] where [cardNumber] = '" + cardNumber + "'";

object obj1 = DataService.ExecuteScalar(cmd1);

float money = float.Parse(obj1.ToString());

if (money >= cost)

{

float newBalance = float.Parse(obj1.ToString()) - cost;

string Updatecmd = "update [CreditCards] set [balance]= '" + newBalance + "' where ([cardNumber]= '" + cardNumber + "')";

DataService.ExecuteNonQuery(Updatecmd);

}

else

CanPay = false;

return CanPay;

}

//פעולה הבודקת האם מספר הביטחון שייך לכרטיס

static public bool cvvBelongToCredit(string cardNumber, int cvv)

{

string cmd = "select [cvv] From [CreditCards] where ([cardNumber] = '" + cardNumber + "')";

object obj = DataService.ExecuteScalar(cmd);

if (int.Parse(obj.ToString()) == cvv)

return true;

return false;

}

//פעולה הבודקת האם התוקף שייך לכרטיס

static public bool ExpireDateBelongToCredit(string cardNumber, string validity)

{

string cmd = "select [validity] From [CreditCards] where ([cardNumber] = '" + cardNumber + "')";

object obj = DataService.ExecuteScalar(cmd);

if (obj.ToString() == validity)

return true;

return false;

}

//פעולה הבודקת האם הכרטיס שייך לחברה שהתקבלה

static public bool CompanyBelongToCredit(string cardNumber, string company)

{

string cmd = "select [company] From [CreditCards] where [cardNumber] = '" + cardNumber + "'";

object obj = DataService.ExecuteScalar(cmd);

if (obj.ToString() == company)

return true;

return false;

}

//פעולה הבודקת האם הכרטיס שייך לבעל תעודת הזהות שהוכנסה

static public bool IDBelongToCredit(string cardNumber, long Id)

{

string cmd = "select [Id] From [CreditCards] where [cardNumber] = '" + cardNumber + "'";

object obj = DataService.ExecuteScalar(cmd);

if (long.Parse(obj.ToString()) == Id)

return true;

return false;

}

//פעולה המחזירה את כל חברות האשראי הקיימות בדטה בייס

public static DataSet AllCompanies()

{

string cmd = "select distinct [Company] from [CreditCards]";

DataSet ds = DataService.GetDataSet(cmd, "Companies");

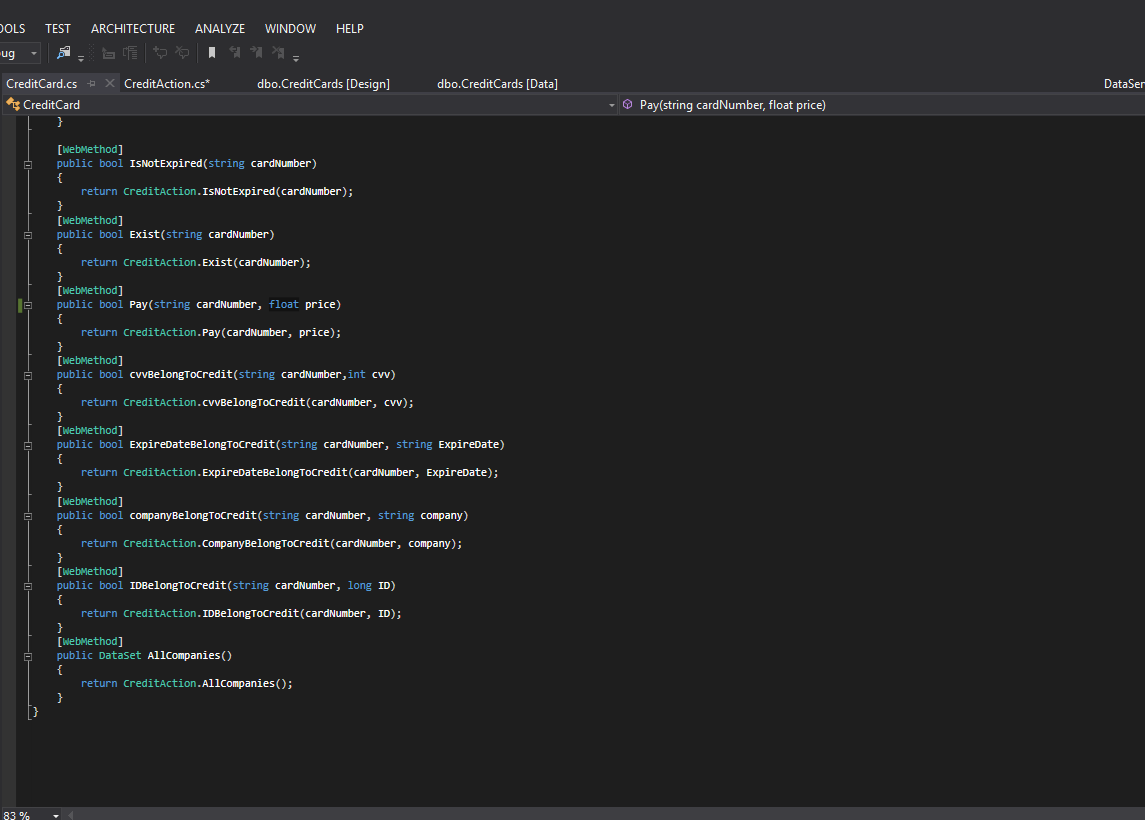
if (ds.Tables["Companies"].Rows.Count == 0)

return null;

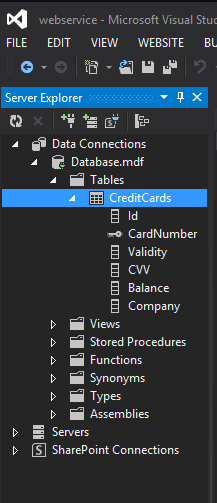
return ds;

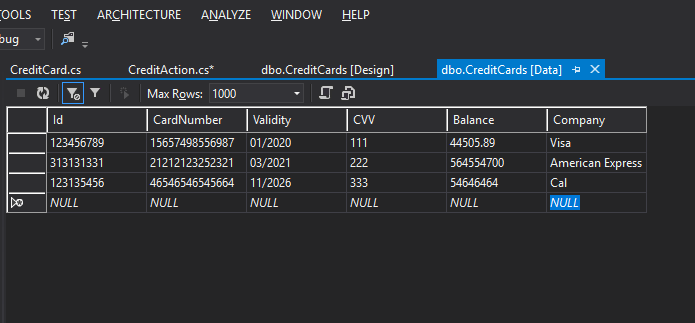
}

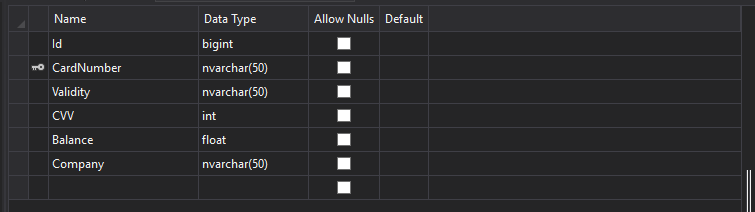
### }



## מבנה בסיס הנתונים של שירותי הרשת







## מפת האתר

דוגמא:

להלן מוצג תרשים הזרימה האפשרי באתר:

Log In

Register

Delete user

Log Out

Global Score

Store

Menu

## מדריך האפליקציה למשתמש

Upload Image

Upload Ship

Ship Manager

Manager Page

My Player

Game

My Profile

Game Over

Buy Points

**האפשרויות הפתוחות בפני משתמש מזדמן:**

LogIn.aspx דף ההתחברות

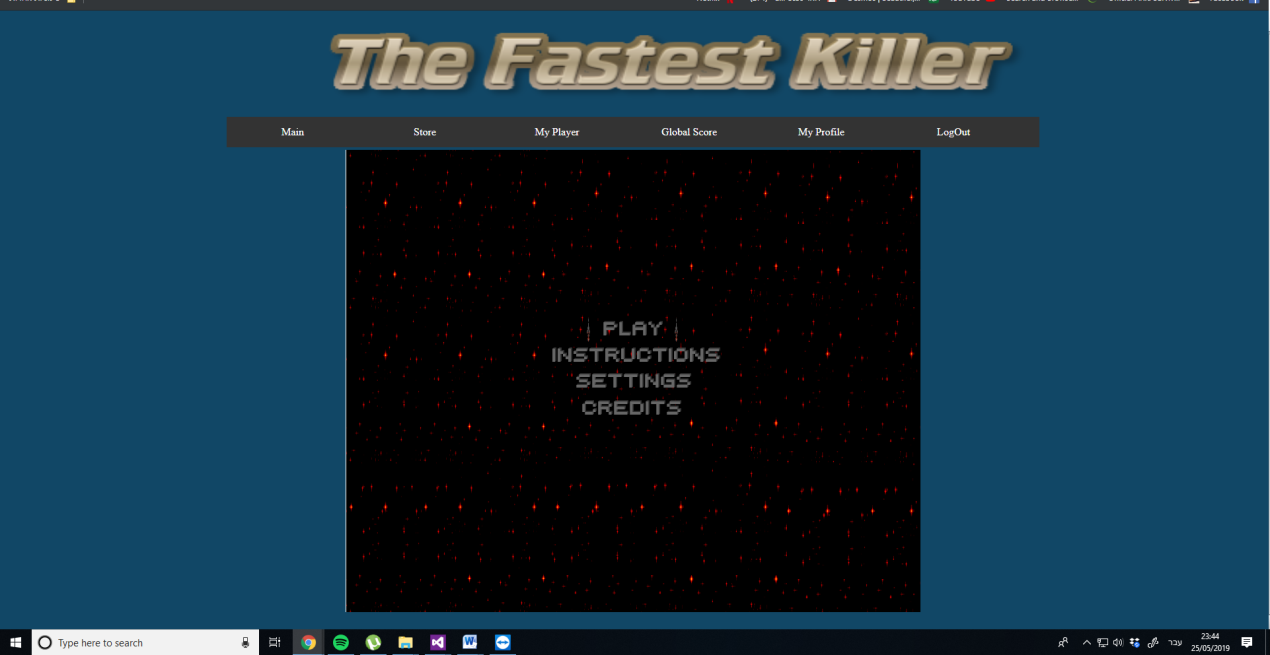
## 

## Registration.aspx דף ההרשמה

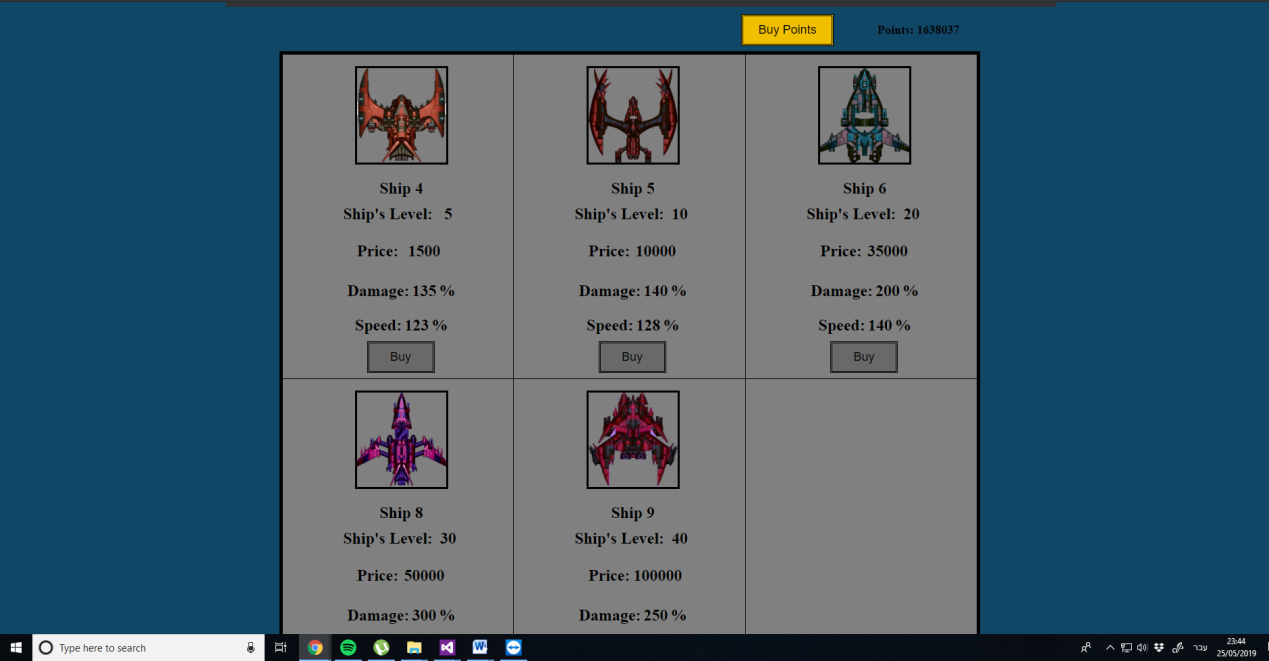
## C:\Users\Adi\Dropbox\Screenshots\Screenshot 2019-05-25 23.43.54.png

## האפשרויות הפתוחות בפני משתמש רשום:

Menu2.aspx



Store1.aspx חנות החלליות



## GlobalScore.aspx טבלת Top 10



## 

## MyProfile.aspx דף בחירת תמונת פרופיל

משתמש יכול לבחור תמונת פרופיל מהתמונות שהרוויח

## C:\Users\Adi\Dropbox\Screenshots\Screenshot 2019-05-25 23.53.11.png

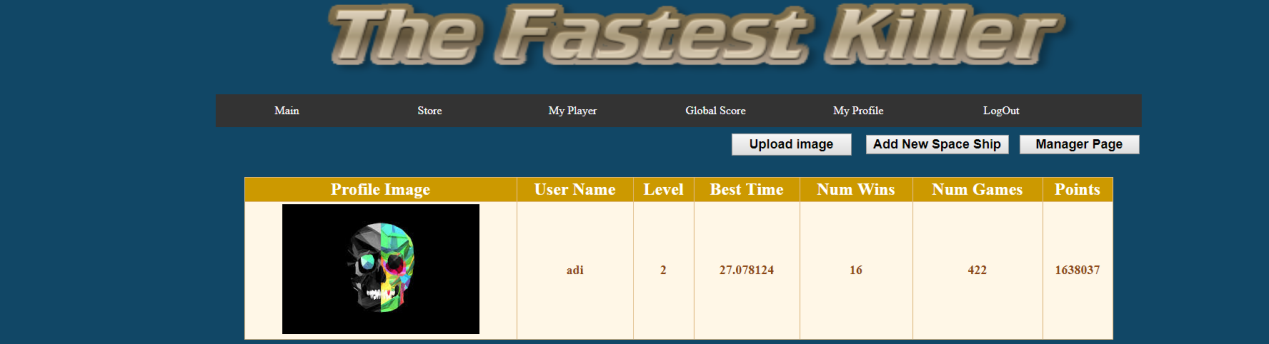
MyPlayer.aspx

דף השחקן ובו יכול השחקן לבחור את חללית המשחק לשדרג אותה



## מדריך האפליקציה למנהל

MyPlayer.aspx דף השחקן (למנהל)



UploadProfileImages.aspx דף ההעלאת תמונות פרופיל

## C:\Users\Adi\Dropbox\Screenshots\Screenshot 2019-05-25 23.45.27.png

## InsertShipPage.aspx דף ההעלת ספינה חדשה

## C:\Users\Adi\Dropbox\Screenshots\Screenshot 2019-05-25 23.45.30.png

## ManagerPage.aspx דף מנהל הספינות בו יכול לשנות את פרטי הספינות

## C:\Users\Adi\Dropbox\Screenshots\Screenshot 2019-05-25 23.45.35.png

## נספחים

במהלך הפרוייקט נדרשתי ללמוד js כדי לבנות את המשחק ונעזרתי בהמון אתרים וסרטונים מכיוון שאיני יכול לציין את כולם ארשום את האתרים העיקריים

<https://www.w3schools.com/graphics/game_intro.asp>

<https://www.youtube.com/watch?v=3EMxBkqC4z0>

<https://www.w3schools.com/graphics/game_canvas.asp>

<https://www.youtube.com/watch?v=gUJiTBIXcMI>

וכדי להשתמש ב css למדתי מהאתר : <https://www.w3schools.com/css/default.asp>

## ReadMe

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

שם משתמש : Test

סיסמה: 123456Test

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

שם משתמש : adi

סיסמה : 123