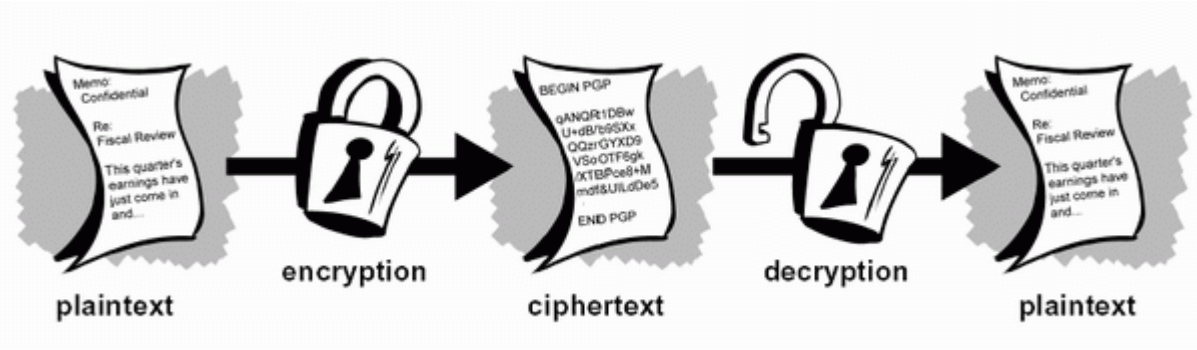
فصل سوم :

رمزنگاری اطلاعات در ASP.NET



مقدمه :

یکی از توانایی های جالب دات نت فریم ورک که در ASP.NET نیز به سادگی قابل استفاده است مبحث Cryptography (کریپتوگرافی / رمزنگاری) می باشد. در این فصل قصد مطالعه ی عمیق الگوریتم های مربوطه را نداریم ، چون صرفا بررسی الگوریتم های موجود رمزنگاری می تواند دوره ای کامل و بسیار پیشرفته به همراه ریاضیاتی بسیار سطح بالا باشد. در این فصل نحوه ی استفاده از توانایی ها و امکانات موجود در این زمینه را به صورت کاربردی بررسی خواهیم کرد.

کریپتوگرافی:

دات نت فریم ورک شامل توابع رمزنگاری برای کد گذاری اطلاعات ، امضای دیجیتال ، Hashing و تولید اعداد تصادفی می باشد. الگوریتم های رمزنگاری که در آن پشتیبانی می شوند شامل : رمز نگاری نامتقارن مانند RSA و DSA و رمزنگاری متقارن همانند DES ، AES ، Triple DES ، RC2 و الگوریتم های Hash مانند MD5 ، SHA1 ، SHA256 ، SHA384 و SHA512 است. طرز کار آن بر مبنای مدل Stream می باشد . بنابراین جریانی از داده ها برای مثال از یک فایل به شیء کدگذار فرستاده شده و نهایتا پس از انجام عملیات، جریان رمزگذاری شده خروجی کار خواهد بود.

این الگوریتم های یاد شده در فضای نام System.Security.Cryptography موجود هستند. در ASP کلاسیک مبحث کد گذاری توسط کامپونت های Com تهیه شده از منبع مختلف انجام می شود اما در اینجا کار بسیار ساده گردیده و دات نت فریم ورک پشتیبانی کاملی را از آن انجام می دهد.

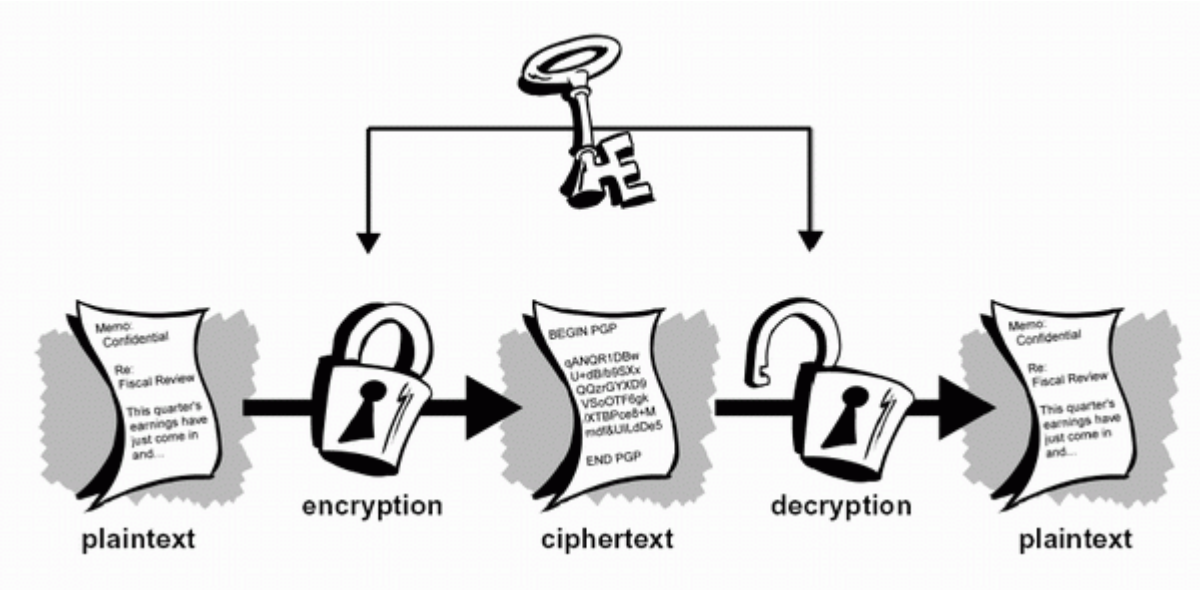
مبانی کریپتوگرافی:

به صورت عمومی انواع الگوریتم های کد گذاری به دو نوع متقارن و غیرمتقارن تقسیم می شوند. در کدگذاری متقارن کلید مخفی و یا همان پسورد برای کد گذاری داده ها بکار می رود. برای رمزگشایی نیز دقیقا همان کلید مخفی باید بکار برده شود. نمونه هایی از این نوع الگوریتم ها شامل DES و RC2 می شود (شکل ۱).

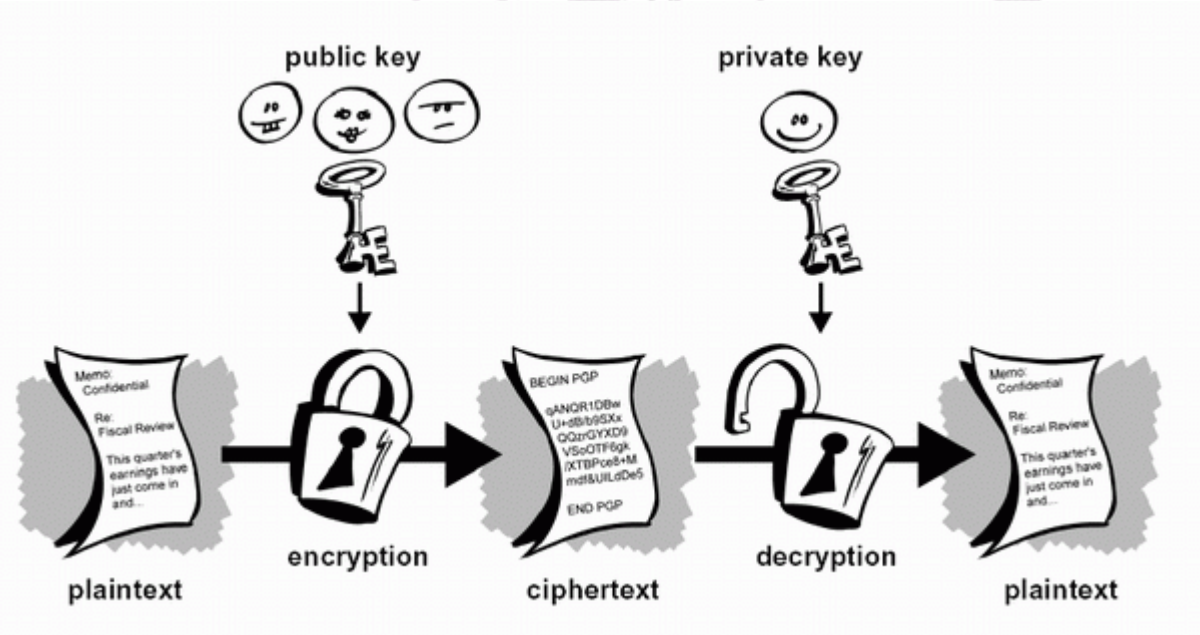
الگوریتم های نامتقارن از یک جفت کلید به هم مربوط برای رمزگذاری و رمزگشایی استفاده می کنند. یکی از این کلیدها کلید عمومی نام می گیرد و دیگری کلید مخفی. داده های کد گذاری شده بوسیله ی یک کلید عمومی تنها بوسیله ی کلید مخفی قابل رمزگشایی هستند و برعکس. مثالی از آن الگوریتم RSA است (شکل ۲).

یکی دیگر از کامپونت های کلیدی دیگر در مبحث کریپتوگرافی ، توابع hash هستند. این توابع داده های ورودی را با هر طولی به یک داده ی خروجی با طول ثابت نگاشت می کنند. توابع hash عموما یک طرفه هستند ، یعنی داده های کدگذاری شده توسط آنها برگشت ناپذیر هستند و برای ایجاد خروجی با طول

ثابت و منحصربفرد کاربرد دارند. کاربرد این توابع متغیر است اما عمومی ترین استفاده ی آنها تولید امضای دیجیتال در رمزگذاری توسط کلید عمومی است (شکل ۳).



شکل ۱- سیستم کد گذاری متقارن (کلید خصوصی) .



شکل ۲- سیستم کدگذاری نامتقارن (کلید عمومی).

مثال فصل:

استفاده از الگوریتم MD5 برای رمزگذاری پسوردهای ذخیره شونده در دیتابیس

در اکثر سایت ها هنگام ورود باید ثبت نام کرد و یک ID و Password مناسب را انتخاب نمود. ثبت پسوردهای کاربران در حالت نرمال به صورت غیرکدگذاری شده صورت می گیرد و اگر کسی به سرور ذکر شده رخنه کند می تواند به پسورد تک تک اعضاء بدون هیچ مشکلی دسترسی داشته باشد و یکی از مواردی که هکرهای روانشناس (!) به آن آگاهی کامل دارند این است که عموما افراد از یک پسورد برای کارها و سایت های مختلف در اینترنت استفاده می کنند زیرا به خاطر سپردن پسوردهای مختلف مشکل می باشد. بنابراین نیاز به بالابردن امنیت سایت در اینترنت به شدت احساس می شود.

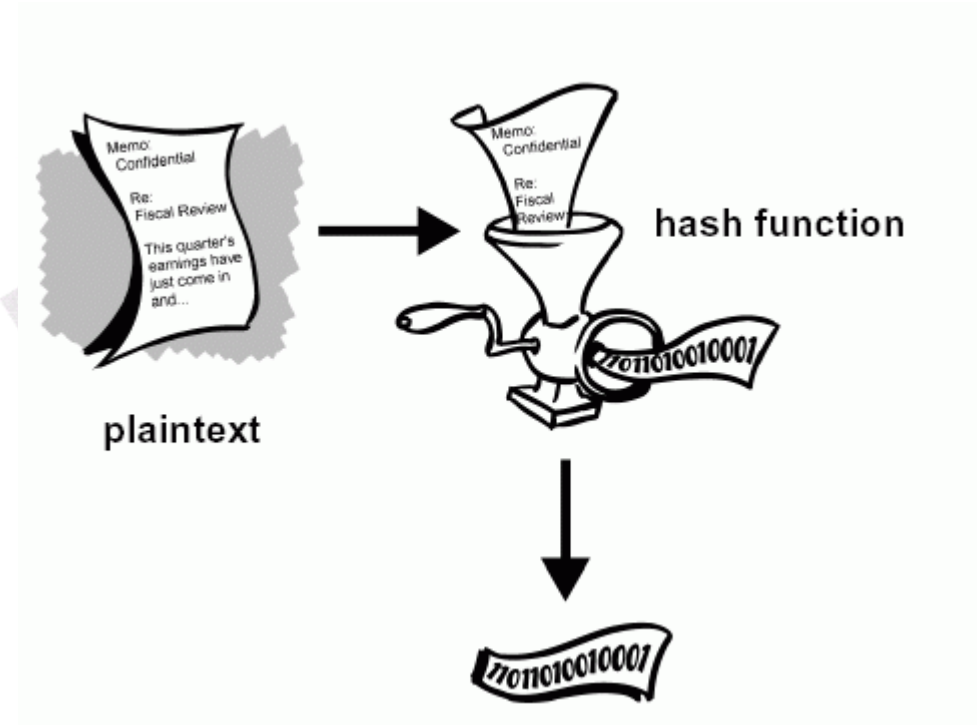
در این مثال قصد داریم از الگوریتم MD5 برای رمزگذاری پسوردهای کاربران هنگام ثبت نام در سایت استفاده کنیم. در این حالت حتی مدیر سایت که به دیتابیس دسترسی تام دارد نیز نمی تواند پسورد واقعی کاربران را حدس بزند.

خلاصه ای در مورد الگوریتم MD5 :

همانطور که گفته شد دو نوع رمزگذاری کلی وجود دارد: رمزگذاری یک طرفه و رمزگذاری دو طرفه. رمزگذاری دوطرفه معمولترین نوع کدگذاری است. در این حالت یک متن عادی (plain-text) دریافت شده و سپس کدگذاری می شود. سپس می توان روش عکس را پیمود و متن را رمزگشایی کرد. برای مثال استفاده از این روش در سایت های تجارت الکترونیکی مناسب می باشد. اگر کسی شماره ی کارت اعتباری خودش را غیررمزنگاری شده بفرستد افرادی که از وسایل monitoring داده ها استفاده می نمایند به راحتی می توانند از آن سوء استفاده کنند. اما اگر به صورت رمزگذاری شده فرستاده شود و سپس سرور آنرا رمزگشایی نموده و استفاده کند احتمال سوء استفاده از آن تقریبا به صفر خواهد رسید.

اما الگوریتم های یک طرفه صرفا داده ها را کد گذاری می کند و هیچ راهی برای رمزگشایی داده های کدگذاری شده توسط آنها وجود ندارد. در نگاه اول به نظر می رسد که این روش بی مصرف است! اما چرا گاهی از اوقات ما می خواهیم داده ها را طوری رمزگذاری کنیم که دیگر قادر به رمز گشایی آنها نباشیم؟! دقیقا موضوع مثال ما به این حالت مربوط می شود که در مورد آن توضیح داده شد.

الگوریتم MD5 مثالی است از الگوریتم های رمزنگاری یک طرفه. این الگوریتم رشته ای را با هر طولی را به یک رشته ی رمز نگاری شده با طول ثابت (۱۶ بایت) نگاشت می کند. دو خاصیت مهم این الگوریتم آن است که دو رشته ی متفاوت ورودی به آن شکل کدگذاری شده ی یکسانی نخواهند داشت (منحصربفرد هستند) و هر داده ی مفروضی همواره به یک رشته ی کدگذاری شده ی یکسان نگاشت می شود (بنابراین اگر کسی یک پسورد را وارد کند همیشه شکل کد گذاری شده ی آن ثابت و منحصربفرد است).



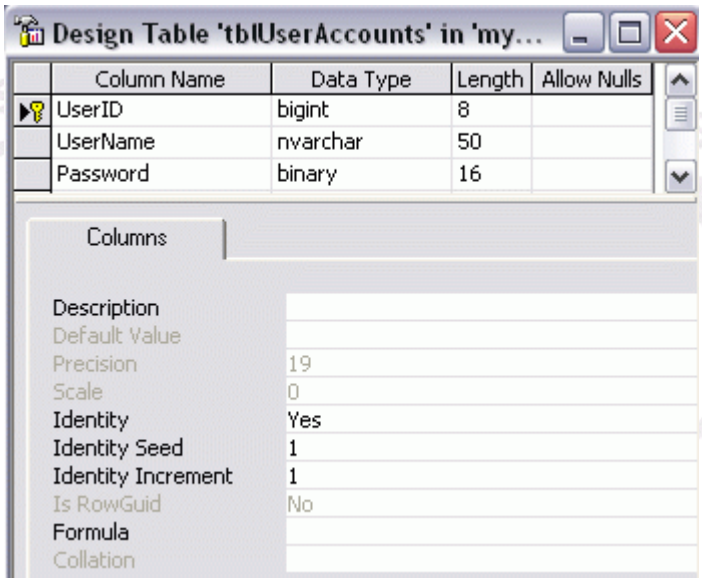
شکل ۳- نحوه ی عملکرد الگوریتم های Hash .

برای استفاده از MD5 در فضای نام System.Security.Cryptography از کلاسی به نام MD5CryptoServiceProvider استفاده می گردد. متد اصلی این کلاس ComputeHash می باشد که ورودی آن آرایه ای از bytes می باشد (رشته ای که باید رمزگذاری شود) و یک آرایه از بایت ها را نیز بر می گرداند که همان داده ی رمزگذاری شده است. عموما ما می خواهیم یک رشته را رمزگذاری کنیم بنابراین باید آنرا به آرایه ای از بایت ها قابل استفاده برای تابع تبدیل نماییم. این تبدیل با استفاده از UTF8Encoding قابل انجام است.

استفاده از MD5 برای رمزگذاری پسوردها و ذخیره کردن آنها

در اغلب دیتابیس ها جدولی به شبیه به UserAccounts وجود دارد با فیلدهایی به نامهای UserName و Password . عموما نوع هر فیلد nvarchar انتخاب می شود. اما برای بحث رمز گذاری بهتر است نوع فیلد پسورد را از نوع binary به طول ۱۶ انتخاب کنیم. زیرا پسورد کاربران با هر طولی تبدیل به آرایه ای از بایتها با ۱۶ المان خواهد شد.

ابتدا در دیتابیس MyTestDB که در SQL-Server ایجاد خواهید نمود جدول زیر را خلق کنید :



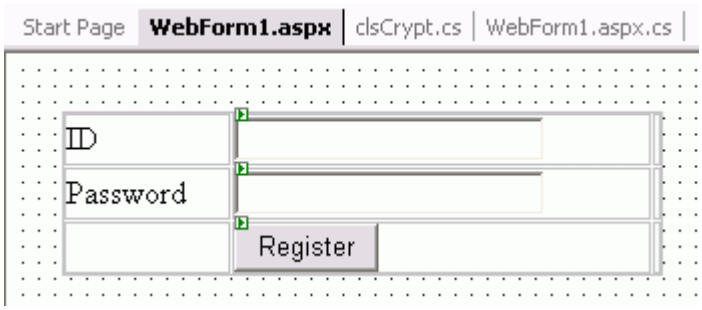
شکل ۴– ایجاد جدول tblUserAccounts در دیتابیس MyTestDB .

یک پروژه ی جدید باز کنید و در ادامه از منوی پروژه یک کلاس جدید را به نام clsCrypt.cs به برنامه اضافه نموده و کد زیر را در آن بنویسید:

از این کد برای محاسبه ی MD5Hash در برنامه استفاده خواهیم کرد :

```
using System;
using System.Security.Cryptography;
using System.Text;

namespace ch03
{
	/// <summary>
	/// Summary description for clsCrypt.
	/// </summary>
	public class clsCrypt
	{
		public clsCrypt()
		{
		}

		public byte[] computeMD5Hash(string strPlainText)
		{
		  //The array of bytes that will contain
		  //the encrypted value of strPlainText
		  byte[] hashedDataBytes = new byte[16];

		  //The encoder class used to convert
		  //strPlainText to an array of bytes
		  UTF8Encoding encoder = new UTF8Encoding();

		  //Create an instance of the MD5CryptoServiceProvider class
		MD5CryptoServiceProvider md5Hasher = new MD5CryptoServiceProvider();

		//Call ComputeHash, passing in the plain-text string as an array of bytes
		 //The return value is the encrypted value, as an array of bytes
		 hashedDataBytes = md5Hasher.ComputeHash(encoder.GetBytes(strPlainText));
		 return hashedDataBytes;

		}
	}
}
```

سپس فرم ظاهر برنامه را به شکل زیر طراحی کنید:

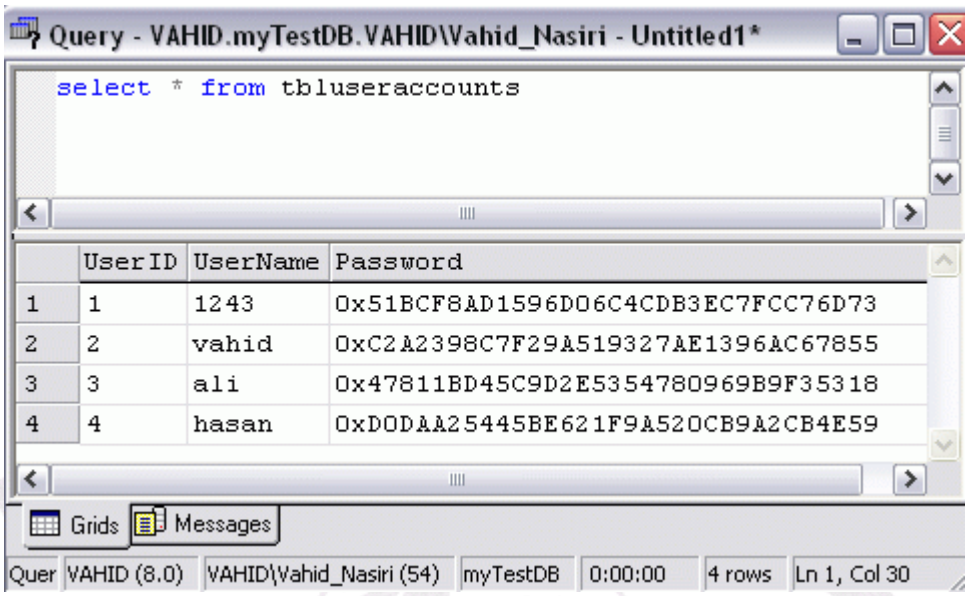


شکل ۵- طراحی ظاهر فرم مثال فصل.

با استفاده از کد زیر می توان عملیات مورد نظر را انجام داد:

```
private void btnAdd_Click(object sender, System.EventArgs e)
{

      //1. Create a connection
      string strConnString  =
            "server=localhost ; uid=sa; pwd=; database=MyTestDB";
      SqlConnection objConn = new SqlConnection(strConnString);


      //2. Create a command object for the query
      string strSQL =
            "INSERT INTO tblUserAccounts(Username,Password) " +
            "VALUES(@Username, @Password)" ;
      SqlCommand objCmd = new SqlCommand(strSQL, objConn);


      //3. Create parameters

      SqlParameter paramUsername = new SqlParameter(
            "@Username", SqlDbType.NVarChar , 50);
      paramUsername.Value = txtID.Text;
      objCmd.Parameters.Add(paramUsername);


      //Encrypt the password
      clsCrypt cr = new clsCrypt();
      byte[] hashedBytes = new byte[16]   ;
      hashedBytes = cr.computeMD5Hash (txtPass.Text);


      SqlParameter paramPwd = new
            SqlParameter("@Password", SqlDbType.Binary, 16);
      paramPwd.Value = hashedBytes;
      objCmd.Parameters.Add(paramPwd);


      //Insert the records into the database
      objConn.Open();
      objCmd.ExecuteNonQuery();
      objConn.Close();

      //clear
      txtID.Text ="";

      //Redirect user to confirmation page...

}
```

و در این شکل با استفاده از Query analyzer نمایی از پسوردهای ذخیره شد را می توان ملاحظه نمود:



شکل ۶– آمار کامل کاربران ثبت شده در دیتابیس.

بدیهی است که حتی اگر در خوشبینانه ترین حالت مدیر سایت به دیتابیس نگاه کند از پسوردها چیزی سر در نخواهد آورد و از آنها نمی توان پسوردهای اصلی رمزگشایی شده را استخراج نمود.

استقاده از MD5 برای بررسی Authentication کاربرها همانند الگوریتم آن ، یک طرفه است و هیچ راهی برای استخراج پسورد واقعی از آن وجود ندارد ، شاید این سوال پرسیده شود که چگونه می توان از آن استفاده کرد؟

برای پاسخ به این سوال می توان به خواص ذاتی MD5 اشاره کرد. هر پسوردی فقط و فقط معادل یک رشته ی کدگذاری شده ی منحصربفرد MD5 است.

بنابراین برای تعیین اعتبار یک کاربر می توان پسورد واقعی را هر بار از کاربر دریافت کرد و سپس توسط MD5 آنرا رمزگذاری نمود. اکنون این رشته ی تولید شده را با رشته ی موجود در دیتابیس مقایسه می کنیم (مطابق تابع زیر).

```
bool isUserValid(string userID, string userPass)
{
      //1. Create a connection
      string strConnString  =
      "server=localhost ; uid=sa; pwd=; database=MyTestDB";
      SqlConnection objConn = new SqlConnection(strConnString);

      //2. Create a command object for the query
      string strSQL ="SELECT COUNT(*) FROM UserAccount " +
            "WHERE Username=@Username AND Password=@Password";
      SqlCommand objCmd = new SqlCommand(strSQL, objConn);

      //3. Create parameters

      SqlParameter paramUsername = new SqlParameter("@Username",
                                    SqlDbType.VarChar, 50);
      paramUsername.Value = userID;
      objCmd.Parameters.Add(paramUsername);

      //Encrypt the password
      byte[] hashedDataBytes = new Byte[16];
      clsCrypt cr = new clsCrypt() ;
      hashedDataBytes = cr.computeMD5Hash (userPass);

      SqlParameter paramPwd = new SqlParameter("@Password",
                                 SqlDbType.Binary, 16);
      paramPwd.Value = hashedDataBytes;
      objCmd.Parameters.Add(paramPwd);

      //Insert the records into the database
      objConn.Open();
      int iResults  = (int) objCmd.ExecuteScalar();
      objConn.Close();

      if( iResults == 1 )
            //The user was found in the DB
            return true;
      else
            //The user was not found in the DB
            return false;

}
```

بدیهی است که با استفاده از این روش گزینه ی : " لطفا اینجا کلیک نمایید تا پسورد فراموش شده ی شما را برایتان ایمیل بزنیم " دیگر معنی نخواهد داشت. در این حالت می توان از یک روش دیگر برای داشتن این موضوع در سایت استفاده کرد. برای کاربر یک عدد رندام انتخاب کرده و به او ایمیل می زنیم

و سپس به او می گوییم که با این پسورد جدید به سایت لاگین کرده و در ادامه پسورد ایمیل شده از طرف ما را عوض کند.

برای اینکه روش گفته شده را بسیار مقاوم تر در برابر حملات نمود می توان بجای رمزگذاری پسورد به تنهایی ، جمع دو رشته ی ID و Pass را با MD5 هش کرد. چون همیشه کاربر باید ID خودش را نیز وارد کند و این ID باید در یک جدول ، منحصر بفرد باشد تا کاربری تکراری ثبت نام نکند.

یکی دیگر از کاربردهای این روش ثبت پسورد در یک کوکی است.

نکته :

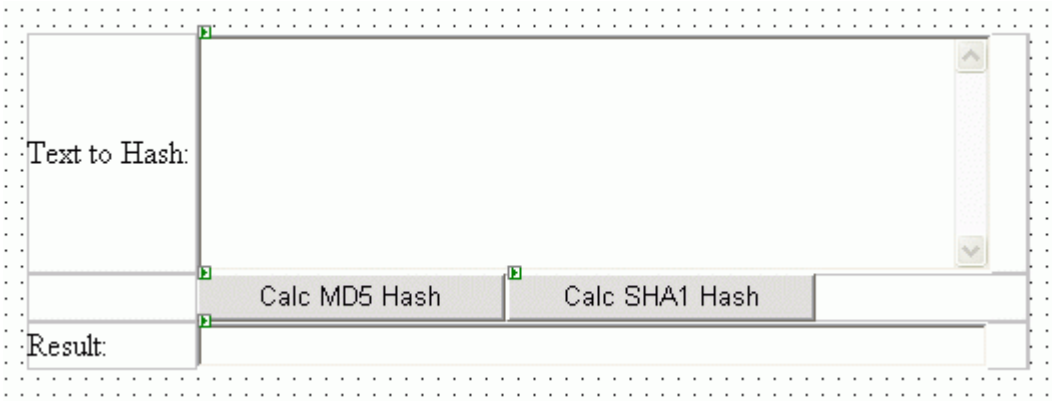
برای هش کردن اطلاعات با استفاده از دو الگوریتم متداول MD5 و یا SHA1 می توان از یک راه میانبر دیگر نیز استفاده کرد. در فضای نام System.Web.Security از متد زیر می توان استفاده کرد:

```
txtRes.Text = FormsAuthentication.HashPasswordForStoringInConfigFile(
                                                    txtInput.Text, "SHA1");
```

آرگومان اول آن متنی است که باید هش شود و آرگومان دوم آن MD5 و یا SHA1 میتواند باشد.

تمرین:

۱- مطابق شکل زیر و نکته ی آخر فصل برنامه ای بنویسید که یک متن را از کاربر گرفته و هش آنرا بر مبنای الگوریتم های MD5 و SHA1 محاسبه کند و نمایش دهد.



۲- مثال فصل را با استفاده از تابع `isUserValid` که همراه سورس برنامه توسعه داده شد ، به صوت زیر تکمیل نمایید:

الف) ایجاد یک صفحه برای رجیستر کردن کاربر جدید (که در برنامه انجام شد).

ب) ایجاد یک صفحه برای لاگین کردن کاربران موجود و سپس نمایش صفحه ی خوش آمدگویی و یا برعکس!

ج) ایجاد یک صفحه برای ایمیل زدن پسورد جدید کاربرانی که پسورد خودشان را فراموش کرده اند.

د) ایجاد یک صفحه با استفاده از دیتاگرید با خاصیت Paging برای نمایش لیست کاربران ثبت نام کرده.

ه) تمام این صفحات باید با استفاده از یک یوزر کنترل که در بالای صفحه شبیه به یک منو قرار می گیرد ارتباط و لینک داشته باشند.