

Set Up a Two Factor Authentication with SMS.

Adding two-factor authentication (2FA) to your web application increases the security of your user's data.

1. First we validate the user with an **email** and **password**
2. Second we validate the user using his or her mobile device, by sending a **one-time verification code SMS**.

Web.config Settings:

```
<add key="DidNumber" value="16505340986"/>
<add key="ApiKey" value="YOUR_API_KEY"/>
<add key="SmsApiBaseUrl"
value="https://api.didforsale.com/didforsaleapi/index.php/api/SMS/send/">
```

Using DIDforSale API for send verification code.

Code:

DID2FA\App_Start\IdentityConfig.cs

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Security.Claims;
using System.Threading.Tasks;
using System.Web;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
using Microsoft.AspNet.Identity.Owin;
using Microsoft.Owin;
using Microsoft.Owin.Security;
using DID2FA.Models;
using System.Web.Configuration;
using System.Net;
using System.IO;

namespace DID2FA
{
    public class EmailService : IIdentityMessageService
    {
        public Task SendAsync(IdentityMessage message)
        {
            // Plug in your email service here to send an email.
            return Task.FromResult(0);
        }
    }
}
```

```

public class SmsService : IIdentityMessageService
{
    public string didNumber =
WebConfigurationManager.AppSettings["DidNumber"].ToString();
    public string apiKey = WebConfigurationManager.AppSettings["ApiKey"].ToString();
    public string smsApiBaseUrl =
WebConfigurationManager.AppSettings["SmsApiBaseUrl"].ToString();

    public Task SendAsync(IdentityMessage message)
    {
        string result = string.Empty;
        //DIDforsale api only accept json data format
        ServicePointManager.SecurityProtocol = SecurityProtocolType.Ssl3 |
SecurityProtocolType.Tls12 | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls;
        var httpRequest = (HttpRequest)WebRequest.Create(smsApiBaseUrl +
apiKey);
        httpRequest.ContentType = "application/json";
        httpRequest.Method = "POST";
        using (var streamWriter = new
StreamWriter(httpRequest.GetRequestStream()))
        {
            var json = "{\"from\": \"" + didNumber + "\", \"to\": \"" +
message.Destination + "\", \"text\": \"" + message.Body + "\"}";
            streamWriter.Write(json);
            streamWriter.Flush();
            streamWriter.Close();
        }
        var httpResponse = (HttpWebResponse)httpRequest.GetResponse();
        using (var streamReader = new StreamReader(httpResponse.GetResponseStream()))
        {
            result = streamReader.ReadToEnd();
        }
        return Task.FromResult(0);
    }
}

// Configure the application user manager used in this application. UserManager is
defined in ASP.NET Identity and is used by the application.
public class ApplicationUserManager : UserManager<ApplicationUser>
{
    public ApplicationUserManager(IUserStore<ApplicationUser> store)
        : base(store)
    {
    }

    public static ApplicationUserManager
Create(IdentityFactoryOptions<ApplicationUserManager> options, IOwinContext context)
    {
        var manager = new ApplicationUserManager(new
UserStore<ApplicationUser>(context.Get<ApplicationDbContext>()));
        // Configure validation logic for usernames
        manager.UserValidator = new UserValidator<ApplicationUser>(manager)
        {
            AllowOnlyAlphanumericUserNames = false,
            RequireUniqueEmail = true
        };
    }
}

```

```

// Configure validation logic for passwords
manager.PasswordValidator = new PasswordValidator
{
    RequiredLength = 6,
    RequireNonLetterOrDigit = true,
    RequireDigit = true,
    RequireLowercase = true,
    RequireUppercase = true,
};

// Configure user lockout defaults
manager.UserLockoutEnabledByDefault = true;
manager.DefaultAccountLockoutTimeSpan = TimeSpan.FromMinutes(5);
manager.MaxFailedAccessAttemptsBeforeLockout = 5;

// Register two factor authentication providers. This application uses Phone
and Emails as a step of receiving a code for verifying the user
// You can write your own provider and plug it in here.
manager.RegisterTwoFactorProvider("Phone Code", new
PhoneNumberTokenProvider<ApplicationUser>
{
    MessageFormat = "Your security code is {0}"
});
manager.RegisterTwoFactorProvider("Email Code", new
EmailTokenProvider<ApplicationUser>
{
    Subject = "Security Code",
    BodyFormat = "Your security code is {0}"
});
manager.EmailService = new EmailService();
manager.SmsService = new SmsService();
var dataProtectionProvider = options.DataProtectionProvider;
if (dataProtectionProvider != null)
{
    manager.UserTokenProvider =
        new
DataProtectorTokenProvider<ApplicationUser>(dataProtectionProvider.Create("ASP.NET
Identity"));
}
return manager;
}
}

// Configure the application sign-in manager which is used in this application.
public class ApplicationSignInManager : SignInManager<ApplicationUser, string>
{
    public ApplicationSignInManager(ApplicationUserManager userManager,
IAuthenticationManager authenticationManager)
        : base(userManager, authenticationManager)
    {
    }

    public override Task<ClaimsIdentity> CreateUserIdentityAsync(ApplicationUser
user)
    {
        return user.GenerateUserIdentityAsync((ApplicationUserManager)userManager);
    }
}

```

```

        public static ApplicationSignInManager
Create(IdentityFactoryOptions<ApplicationSignInManager> options, IOwinContext context)
    {
        return new
ApplicationSignInManager(context.GetUserManager<ApplicationUserManager>(),
context.Authentication);
    }
}

```

Code description:

Fetch required values from web.config.

```

        public string didNumber =
WebConfigurationManager.AppSettings["DidNumber"].ToString();
        public string apiKey = WebConfigurationManager.AppSettings["ApiKey"].ToString();
        public string smsApiBaseUrl =
WebConfigurationManager.AppSettings["SmsApiBaseUrl"].ToString();

```

Send verification code to number via DIDforSale

```

string result = string.Empty;
//DIDforsale api only accept json data format
ServicePointManager.SecurityProtocol = SecurityProtocolType.Ssl3 |
SecurityProtocolType.Tls12 | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls;
var httpRequest = (HttpRequest)WebRequest.Create(smsApiBaseUrl +
apiKey);
httpRequest.ContentType = "application/json";
httpRequest.Method = "POST";
using (var streamWriter = new
StreamWriter(httpRequest.GetRequestStream()))
{
    var json = "{\"from\": \"" + didNumber + "\", \"to\": [\"" +
message.Destination + "\"], \"text\": \"" + message.Body + "\"}";
    streamWriter.Write(json);
    streamWriter.Flush();
    streamWriter.Close();
}
var httpResponse = (HttpWebResponse)httpRequest.GetResponse();
using (var streamReader = new StreamReader(httpResponse.GetResponseStream()))
{
    result = streamReader.ReadToEnd();
}

```

Use above code to send SMS to register number using didforsale api.

```

// Register two factor authentication providers. This application uses Phone and Emails
as a step of receiving a code for verification of the user
// You can write your own provider and plug it in here.

```

```
manager.RegisterTwoFactorProvider("Phone Code", new
PhoneNumberTokenProvider<ApplicationUser>
{
    MessageFormat = "Your security code is {0}"
});
```

Take a sample application in visual studio and enable Two factor authentication. And add a phone number then you need to check above code and update it accordingly.