private static ApplicationDbContext userContext = new ApplicationDbContext();

private static NotesContext db = new NotesContext();

public static void CheckRole(string roleName)

{

var roleManager = new RoleManager<IdentityRole>(new RoleStore<IdentityRole>(userContext));

// Check to see if Role Exists, if not create it

if (!roleManager.RoleExists(roleName))

{

roleManager.Create(new IdentityRole(roleName));

}

}

public static void CheckSuperUser(string role)

{

var userManager = new UserManager<ApplicationUser>(new UserStore<ApplicationUser>(userContext));

var email = WebConfigurationManager.AppSettings["AdminUser"];

var password = WebConfigurationManager.AppSettings["AdminPassWord"];

var userASP = userManager.FindByName(email);

if (userASP == null)

{

CreateUserASP(email, role, password);

return;

}

}

public static void CreateUserASP(string email)

{

var userManager = new UserManager<ApplicationUser>(new UserStore<ApplicationUser>(userContext));

var userASP = new ApplicationUser

{

Email = email,

UserName = email,

};

userManager.Create(userASP, email);

}

public static void CreateUserASP(string email, string roleName)

{

var userManager = new UserManager<ApplicationUser>(new UserStore<ApplicationUser>(userContext));

var userASP = new ApplicationUser

{

Email = email,

UserName = email,

};

userManager.Create(userASP, email);

userManager.AddToRole(userASP.Id, roleName);

}

public static void AddRoleToUser(string email, string roleName)

{

var userManager = new UserManager<ApplicationUser>(new UserStore<ApplicationUser>(userContext));

var userASP = userManager.FindByEmail(email);

if (userASP == null)

{

return;

}

userManager.AddToRole(userASP.Id, roleName);

}

public static void CreateUserASP(string email, string roleName, string password)

{

var userManager = new UserManager<ApplicationUser>(new UserStore<ApplicationUser>(userContext));

var userASP = new ApplicationUser

{

Email = email,

UserName = email,

};

userManager.Create(userASP, password);

userManager.AddToRole(userASP.Id, roleName);

}

public static async Task SendMail(string to, string subject, string body)

{

var message = new MailMessage();

message.To.Add(new MailAddress(to));

message.From = new MailAddress(WebConfigurationManager.AppSettings["AdminUser"]);

message.Subject = subject;

message.Body = body;

message.IsBodyHtml = true;

using (var smtp = new SmtpClient())

{

var credential = new NetworkCredential

{

UserName = WebConfigurationManager.AppSettings["AdminUser"],

Password = WebConfigurationManager.AppSettings["AdminPassWord"]

};

smtp.Credentials = credential;

smtp.Host = WebConfigurationManager.AppSettings["SMTPName"];

smtp.Port = int.Parse(WebConfigurationManager.AppSettings["SMTPPort"]);

smtp.EnableSsl = true;

await smtp.SendMailAsync(message);

}

}

public static async Task SendMail(List<string> mails, string subject, string body)

{

var message = new MailMessage();

foreach (var to in mails)

{

message.To.Add(new MailAddress(to));

}

message.From = new MailAddress(WebConfigurationManager.AppSettings["AdminUser"]);

message.Subject = subject;

message.Body = body;

message.IsBodyHtml = true;

using (var smtp = new SmtpClient())

{

var credential = new NetworkCredential

{

UserName = WebConfigurationManager.AppSettings["AdminUser"],

Password = WebConfigurationManager.AppSettings["AdminPassWord"]

};

smtp.Credentials = credential;

smtp.Host = WebConfigurationManager.AppSettings["SMTPName"];

smtp.Port = int.Parse(WebConfigurationManager.AppSettings["SMTPPort"]);

smtp.EnableSsl = true;

await smtp.SendMailAsync(message);

}

}

public static async Task PasswordRecovery(string email)

{

var userManager = new UserManager<ApplicationUser>(new UserStore<ApplicationUser>(userContext));

var userASP = userManager.FindByEmail(email);

if (userASP == null)

{

return;

}

var user = db.Users.Where(tp => tp.UserName == email).FirstOrDefault();

if (user == null)

{

return;

}

var random = new Random();

var newPassword = string.Format("{0}{1}{2:04}\*",

user.FirstName.Trim().ToUpper().Substring(0, 1),

user.LastName.Trim().ToLower(),

random.Next(10000));

userManager.RemovePassword(userASP.Id);

userManager.AddPassword(userASP.Id, newPassword);

var subject = "Notes Password Recovery";

var body = string.Format(@"

<h1>Taxes Password Recovery</h1>

<p>Yor new password is: <strong>{0}</strong></p>

<p>Please change it for one, that you remember easyly",

newPassword);

await SendMail(email, subject, body);

}

public static string UploadPhoto(HttpPostedFileBase file)

{

// Upload image

string path = string.Empty;

string pic = string.Empty;

if (file != null)

{

pic = Path.GetFileName(file.FileName);

path = Path.Combine(HttpContext.Current.Server.MapPath("~/Content/Photos"), pic);

file.SaveAs(path);

using (MemoryStream ms = new MemoryStream())

{

file.InputStream.CopyTo(ms);

byte[] array = ms.GetBuffer();

}

}

return pic;

}

public void Dispose()

{

userContext.Dispose();

db.Dispose();

}

}