

Membangun Aplikasi Web Menggunakan Entity Framework dan MVC 5: Bagian 4

Menambah GetUserProfile() Method

Buka class "userManager", dan menambahkan metode berikut ini:

```
public UserProfileView GetUserProfile(int userID) {
    UserProfileView UPV = new UserProfileView();
    using (DemoDBEntities db = new DemoDBEntities()) {
        var user = db.SYSUsers.Find(userID);
        if (user != null) {
            UPV.SYSUserID = user.SYSUserID;
            UPV.LoginName = user.LoginName;
            UPV.Password = user.PasswordEncryptedText;

            var SUP = db.SYSUserProfiles.Find(userID);
            if (SUP != null) {
                UPV.FirstName = SUP.FirstName;
                UPV.LastName = SUP.LastName;
                UPV.Gender = SUP.Gender;
            }

            var SUR = db.SYSUserRoles.Find(userID);
            if (SUR != null) {
                UPV.LOOKUPRoleID = SUR.LOOKUPRoleID;
                UPV.RoleName = SUR.LOOKUPRole.RoleName;
                UPV.IsRoleActive = SUR.IsActive;
            }
        }
    }
    return UPV;
}
```

Metode di atas mendapat informasi pengguna tertentu dari database dengan melewati SYSUserID sebagai parameter. Metode mengembalikan tipe UserProfileView yang memegang beberapa sifat dari tabel yang berbeda.

Menambah EditProfile() Action Method

Sekarang, buka class "HomeController" dan menambahkan action method berikut ini:

```
[Authorize]
public ActionResult EditProfile()
{
    string loginName = User.Identity.Name;
    UserManager UM = new UserManager();
    UserProfileView UPV = UM.GetUserProfile(UM.GetUserID(loginName));
    return View(UPV);
}

[HttpPost]
[Authorize]
public ActionResult EditProfile(UserProfileView profile)
{
    if (ModelState.IsValid)
    {
        UserManager UM = new UserManager();
        UM.UpdateUserAccount(profile);

        ViewBag.Status = "Update Sucessful!";
    }
    return View(profile);
}
```

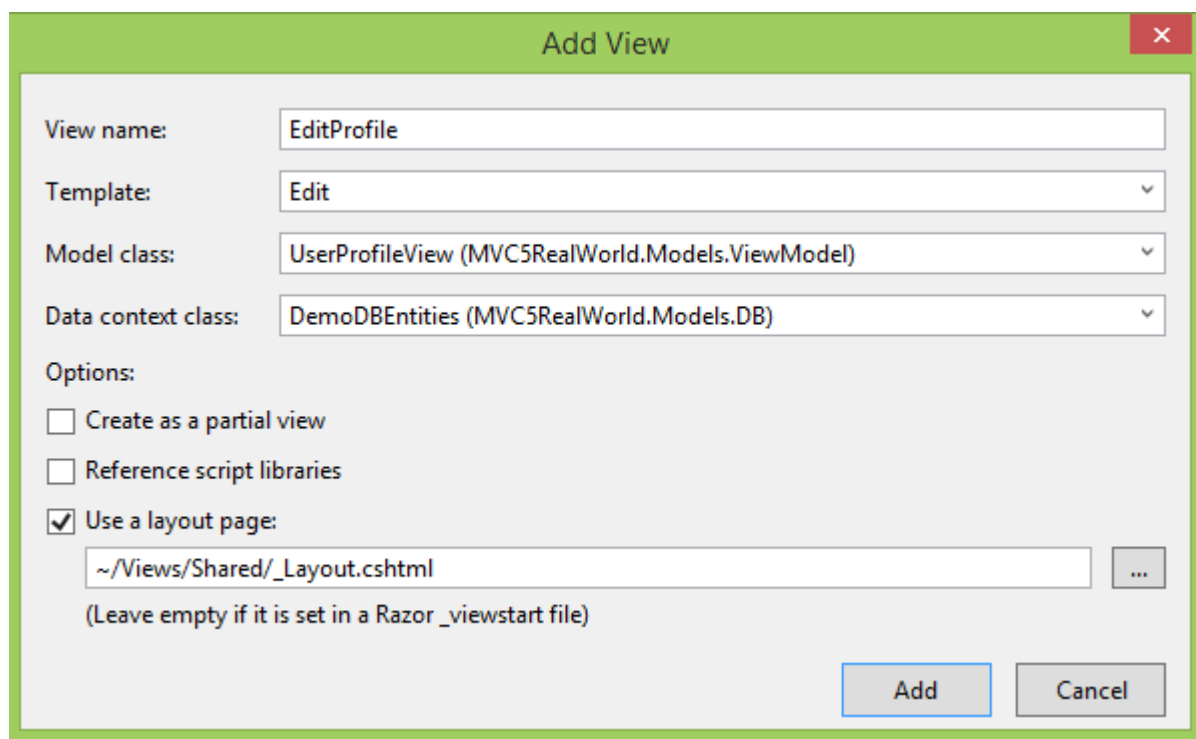
Kode di atas, terdiri dari dua action metode. Metode pertama EditProfile () akan dipanggil, setelah halaman diminta dan dimuat ke browser. Apa yang dilakukannya adalah untuk mendapatkan data profil pengguna dengan memanggil metode GetUserProfile () dan melewati SYSUserID sebagai parameter. Yang kedua adalah metode yang meng-overload yang akan dipanggil saat permintaan POST, yaitu ketika anda menekan tombol untuk menyimpan data. Apa yang dilakukannya adalah bahwa cek pertama untuk keabsahan bidang (jika mereka sah dan tidak kosong), kemudian memanggil metode UpdateUserAccount () dan melewati model

UserProfileView dari View ke metode tersebut. Jika Anda masih ingat dari artikel seri sebelumnya, metode UpdateUserAccount () adalah di mana ia mengeksekusi penghematan aktual data ke database Anda.

Perhatikan bahwa kedua action method diberi [Authorize] atribut untuk memastikan bahwa kedua metode hanya akan diakses oleh para pengguna.

Menambah View

Langkah berikutnya adalah untuk menghasilkan View untuk halaman profil. Untuk melakukan ini, klik kanan pada metode EditProfile () dan pilih "Add View". Dalam Tambah View dialog, menyediakan bidang yang diperlukan seperti yang ditunjukkan pada gambar di bawah:



Gambar 1: Add View dialog

Model class "UserProfileView". lalu, klik Add.

Visual Studio akan menghasilkan semua kontrol di View, berdasarkan bidang Anda didefinisikan dari Model Anda (UserProfileView). Ini berarti bahwa hal itu juga akan menghasilkan bagian yang tidak diinginkan, seperti, LOOKUPRoleID dan IsRoleActive. Memberi daftar drop-down untuk menampilkan bagian Gender. Jadi, pastikan untuk memperbarui kode HTML yang dihasilkan. Ini akan terlihat seperti berikut:

```

@model Asp_mvc_2.Models.ViewModel.UserProfileView

@{
    ViewBag.Title = "EditProfile";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>Edit Your Profile</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <hr />
        <span class="alert-success">@ViewBag.Status</span>
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        @Html.HiddenFor(model => model.SYSUserID)

        <div class="form-group">
            @Html.LabelFor(model => model.RoleName, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.DisplayFor(model => model.RoleName)
                @Html.ValidationMessageFor(model => model.RoleName, "", new { @class
= "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.LoginName, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.LoginName, new { htmlAttributes = new
{ @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.LoginName, "", new { @class
= "text-danger" })
            </div>
        </div>
    </div>

```

```

        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.Password, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.Password, new { htmlAttributes = new {
@class = "form-control" } })
            @Html.ValidationMessageFor(model => model.Password, "", new { @class
= "text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.FirstName, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.FirstName, new { htmlAttributes = new
{ @class = "form-control" } })
            @Html.ValidationMessageFor(model => model.FirstName, "", new { @class
= "text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.LastName, htmlAttributes: new { @class =
"control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.LastName, new { htmlAttributes = new {
@class = "form-control" } })
            @Html.ValidationMessageFor(model => model.LastName, "", new { @class
= "text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.Gender, htmlAttributes: new { @class =
"control-label col-md-2" })

```

```

        <div class="col-md-10">
            @Html.DropDownListFor(model => model.Gender, new List<SelectListItem>
{
                new SelectListItem { Text="Male", Value="M" },
                new SelectListItem { Text="Female", Value="F" }
            }, new { @class = "form-control" })
        </div>
    </div>

    <div class="form-group">
        <div class="col-md-offset-2 col-md-10">
            <input type="submit" value="Save" class="btn btn-default" />
        </div>
    </div>
</div>

<div>
    @Html.ActionLink("Back", "Welcome")
</div>

```

Tambahkan kode berikut pada “Welcome.cshtml”.

```
@Html.ActionLink("Edit Profile", "EditProfile", "Home")
```

kode di atas, tidak lain adalah sebuah link ke halaman Edit Profile, sehingga ketika Anda login, Anda dapat dengan mudah menavigasi ke halaman profil Anda dan mulai memodifikasi data yang.

Output

Now, try to build your own code and run your application. The output should look similar to the figure below:

Application name

Edit Your Profile

RoleName
Member

Login ID

Password

First Name

Last Name

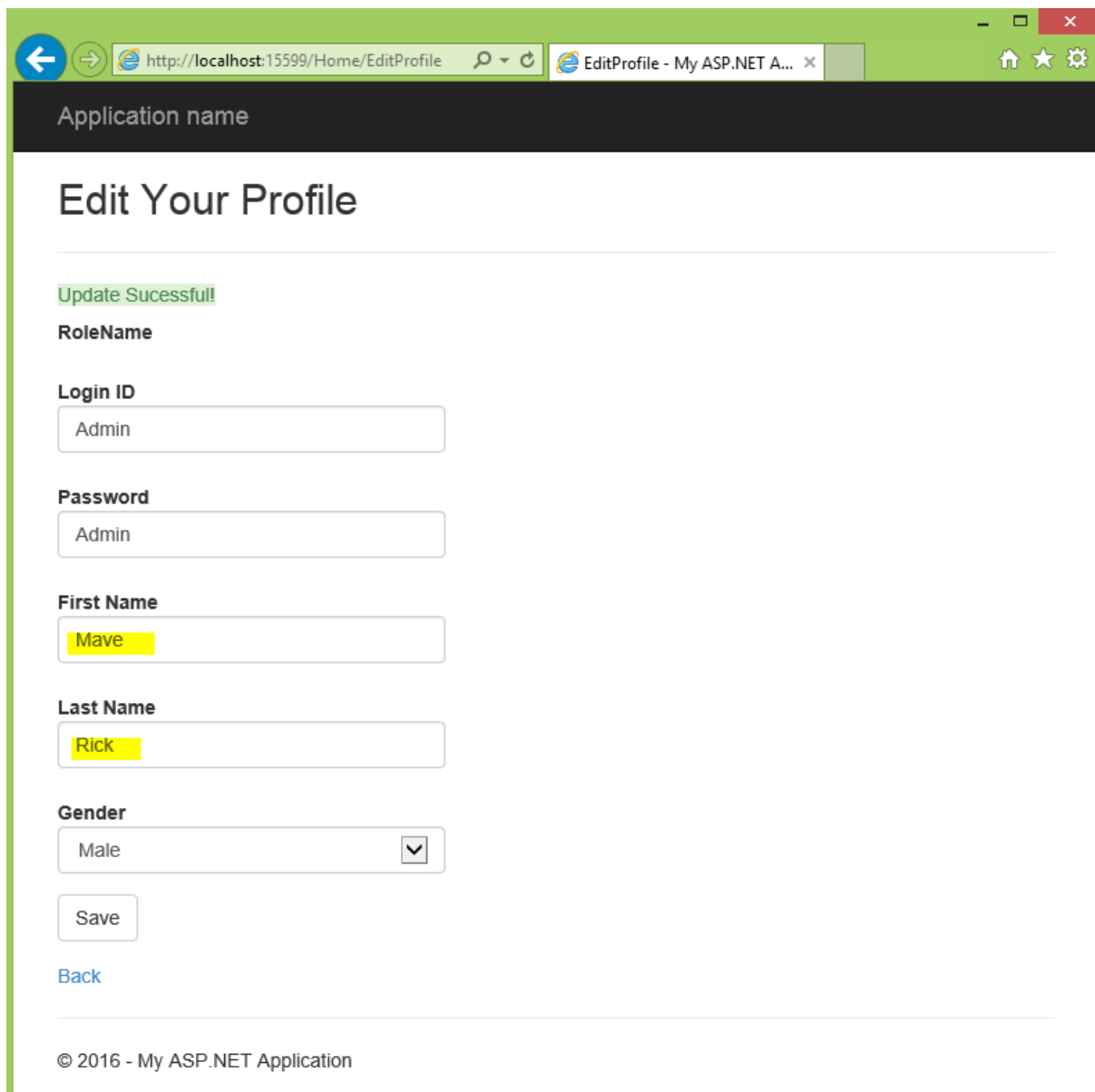
Gender
 ▼

[Back](#)

© 2016 - My ASP.NET Application

Figure 2: Halaman Edit Profile

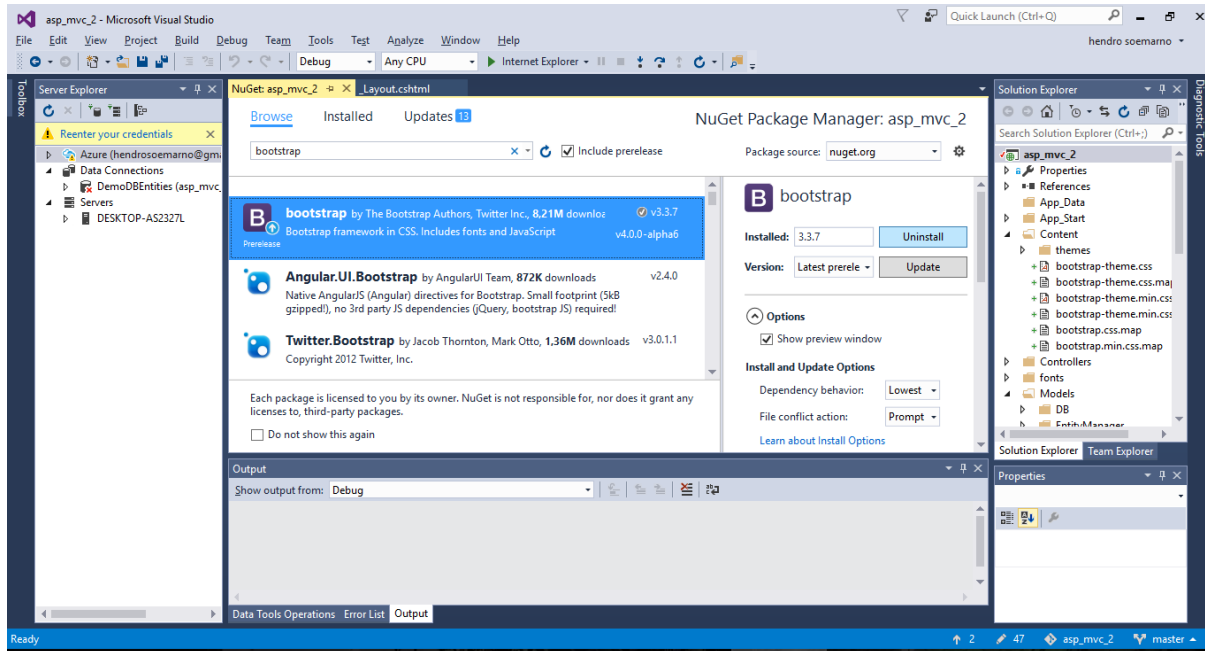
Setelah modifikasi data



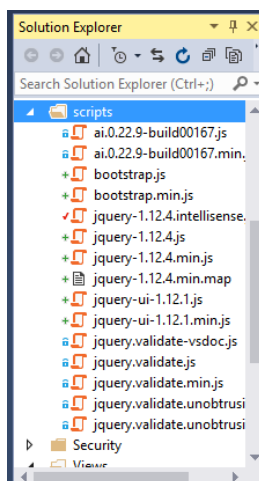
Gambar 3: Setelah Sukses Update Data

Bonus : Menambahkan menu Drop Down

Instal bootstrap melalui klik kanan pada project dan pilih Manage NuGet Package, lalu pada browse, seperti gambar ketik bootstrap, setelah muncul, klik install.



Maka tampilan folder script pada solution explorer akan seperti berikut ini



Lalu modifikasi file _Layout.cshtml seperti berikut ini :

```
<!DOCTYPE html>  
<html>
```

```

<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - My ASP.NET Application</title>
    <link href="~/Content/Site.css" rel="stylesheet" type="text/css" />
    <link href="~/Content/bootstrap.min.css" rel="stylesheet" type="text/css" />
    <script src="~/Scripts/modernizr-2.6.2.js"></script>
    <script src="~/Scripts/jquery-1.12.4.min.js"></script>
    <script src="~/Scripts/jquery-ui-1.12.1.min.js"></script>
    <link href="~/Content/themes/base/all.css" rel="stylesheet" />
</head>
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-
target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                @Html.ActionLink("Application name", "Index", "Home", new { area = "" },
new { @class = "navbar-brand" })
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li class="dropdown">@Html.ActionLink("Home", "Index", "Home")</li>

                    <!-- DropDown example -->
                    <li class="dropdown">
                        <a href="#" class="dropdown-toggle" id="dropdownCommonMenu" data-
toggle="dropdown">DropDown</a>
                        <ul class="dropdown-menu" role="menu" aria-
labelledby="dropdownCommonMenu">
                            <li role="menuitem" class="nav-header">Header 1</li>
                            <li role="menuitem">@Html.ActionLink("Item 1", "Index",
"Home")</li>
                            <li role="menuitem" class="disabled">@Html.ActionLink("Item 2
(disabled)", "Index", "Home")</li>
                            <li role="menuitem" class="divider"></li>
                            <li role="menuitem" class="nav-header">Header 2</li>
                            <li role="menuitem">@Html.ActionLink("Item 3", "Index",
"Home")</li>
                        </ul>
                    </li>

                </ul>
            </div>
        </div>
    </div>

    <div class="container body-content">
        @RenderBody()
        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
        </footer>
    </div>

```

```
<script src="../../Scripts/bootstrap.min.js"></script>
</body>
</html>
```

Maka tampilan akan menghasilkan gambar berikut ini:

