

SPA AND AJAX

1. Build your Model.

```
public class ContactViewModel
{
    [Required(ErrorMessage = "* Name is required")]
    public string Name { get; set; }

    [DataType(DataType.EmailAddress)]
    [Required(ErrorMessage = "* Email is required")]
    public string Email { get; set; }

    public string Subject { get; set; }

    [Required(ErrorMessage = "* Message is required")]
    [UIHint("MultilineText")]
    public string Message { get; set; }
}
```

2. Add the sensitive data necessary to process and send email communication into an external configuration file.
 - a. First, you will right click on the solution and add a folder called configs (if it is not already there). Then you will right click on the configs folder and select Add > XML File. You will name the file AppSecretKeys.config. Into that file, you will add the following code.

```
<?xml version="1.0" encoding="utf-8"?>
<appSettings>
    <!--Added to protect sensitive data-->

    <!--Replace yourdomain.com with your actual domain and extension-->
    <add key="EmailClient" value="mail.yourdomain.com"/>
    <!--Email user on your host-->
    <add key="EmailUser" value="email@yourdomain.com"/>
    <!--Password for the email user on your host-->
    <add key="EmailPass" value="Password"/>
    <!--Email Address to send contact form entries to-->
    <add key="EmailTo" value="email@domain.com"/>
</appSettings>
```

- b. You will then connect this external configuration file to the web.config file in the root of the UI layer by adding the file attribute, as shown below, onto the opening <appSettings> tag.

```
<appSettings file="configs\AppSecretKeys.config">
  <add key="webpages:Version" value="3.0.0.0" />
  <add key="webpages:Enabled" value="false" />
  <add key="ClientValidationEnabled" value="true" />
  <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  <add key="owin:AppStartup" value="IdentitySample.Startup,TestingProject" />
</appSettings>
```

3. Build or Correct the Action in the Home Controller. No get, because the form is embedded in the index.html or the _Layout (pictured Later).

```
[HttpPost]
public JsonResult ContactAjax(ContactViewModel cvm)
{
    //You can make this whatever you want, it will be the body of the message sent
    string body = $"{cvm.Name} has sent you the following message:<br/>" +
        $"{cvm.Message} <strong>from the email address:</strong> {cvm.Email}.";
    //Message Object
    MailMessage mm = new MailMessage(
        //FROM address - email must be on host - creds stored in Web.config
        ConfigurationManager.AppSettings["EmailUser"].ToString(),
        //TO - email doesn't have to be on host - creds stored in Web.config
        ConfigurationManager.AppSettings["EmailTo"].ToString(),
        //email subject
        cvm.Subject,
        //body of the email
        body);

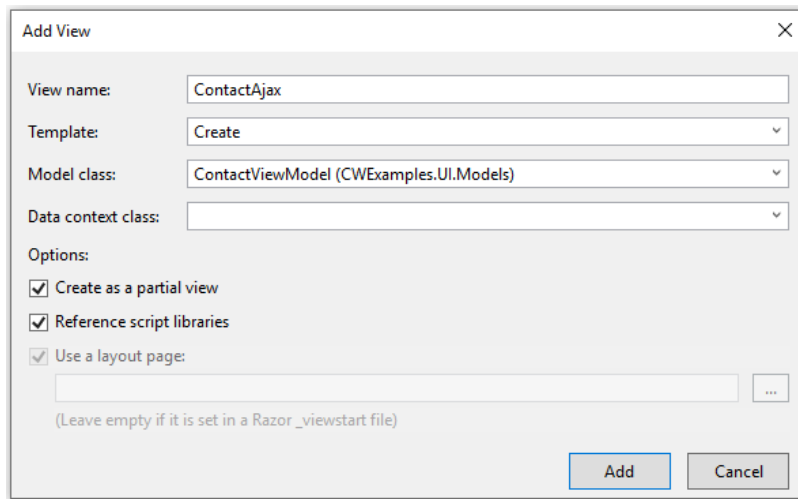
    //allow HTML in email (that is our formatting with br and strong tags above)
    mm.IsBodyHtml = true;
    //you can make the message be designated as high priority
    mm.Priority = MailPriority.High;
    //reply to the Person who filled out the form, not your domain email
    mm.ReplyToList.Add(cvm.Email);

    //configure the mail client - creds stored in web.config
    SmtpClient client = new SmtpClient(ConfigurationManager.AppSettings["EmailClient"].ToString());
    //configure the email credentials using values from web.config
    client.Credentials = new NetworkCredential(ConfigurationManager.AppSettings["EmailUser"].ToString(),
        ConfigurationManager.AppSettings["EmailPass"].ToString());

    try
    {
        //send email
        client.Send(mm);
    }
    catch (Exception ex)
    {
        //log error in ViewBag to be seen by admins
        ViewBag.Message = ex.StackTrace;
    }

    return Json(cvm);
}
```

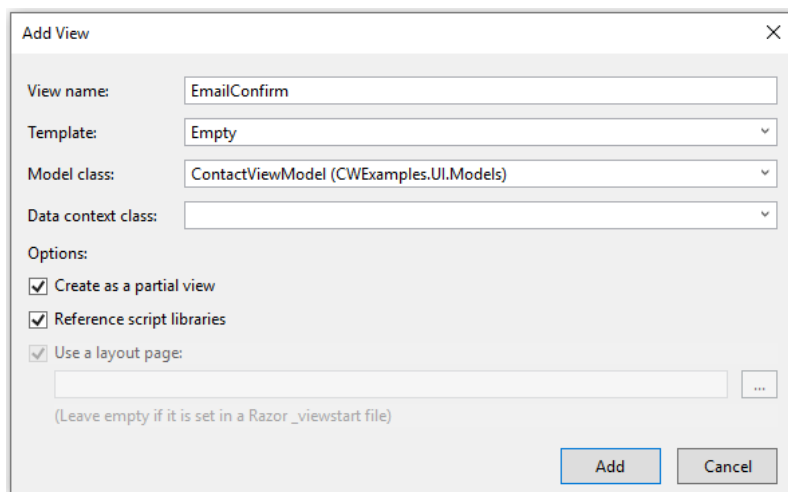
4. Add a Partial View by right clicking the ContactAJAX Action.



5. The only change to be made in the form generated:

```
@using (Html.BeginForm("", "", FormMethod.Post, new {id="ContactCreateForm"}))  
{  
    @Html.AntiForgeryToken()  
}
```

6. Add another partial view of your Email Confirmation. Right click the Home Folder and Add View.



7. Add some HTML to that view to indicate you have received the email:

```
@model CWExamples.UI.Models.ContactViewModel  
  
<div>  
    @if (Model != null)  
    {  
        <h2>Thank you for your message @Model.Name.</h2>  
        <p>We will respond to you at @Model.Email as soon as possible!</p>  
    }  
</div>
```

8. Place the reference to the contact form and the confirmation in the Layout:

```
<div class="container body-content">
  @RenderBody()

  <div id="contactForm">
    @Html.Partial("ContactAjax", new CWExamples.UI.Models.ContactViewModel())
  </div>
  <div id="contactConfirm">
    @Html.Partial("EmailConfirm", new CWExamples.UI.Models.ContactViewModel())
  </div>
</div>
```

9. In the _Layout.cs, add a link to JQuery/AJAX: <https://cdnjs.com/libraries/jquery/>
- Locate: <https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.js>
 - Add to Layout: `<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.js"></script>`
 - Start with this link at the bottom of your javascript links but above the `RenderSection("scripts", false)`
 - You may have to move this around to get it to function.
10. Lastly, in the Layout.cs, add the following AFTER all jQuery links and the `RenderSection(scripts,false)`:

```
<script>
    $("#ContactCreateForm").submit(function (e) {
        var formData = $(this).serializeArray();
        e.preventDefault();
        $.ajax({
            url: '@Url.Action("ContactAjax","Home")',
            type: 'POST',
            data: formData,
            datatype: 'json',
            success: function (data) {
                $("#contactForm").hide();
                //in your style sheet you should have a line to
                //hide this div until this time:
                // #contactConfirm{display:none;}
                $("#contactConfirm").show();
            }
        });
    });
</script>
```