## >Create new project from bitbucket

VS, Bitbucket Extension, Login bitbucket, Clone, select folder

## >.Net 2015 References with yellow triangle

1. Open .csproj

removed the first one:
`<Error Condition="!Exists('..\packages\`

and kept this one:
`<Error Condition="!Exists('..\..\..\packages\`

2. In solution check the target framework version in each projects

3. clear projects and solution. removed References with yellow triangle and reinstall

4. add some new project and delete?

after change, unstage all change. (undo change)

not sure 3-4 is solution

In a MVC solution, first project is [ASP.NET Web Application].
If add a new project, should select [Windows Forms Application],
then delete all forms

## >(localdb)\MSSQLLocalDB location:

connectionString="Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=MAF_dev;
Integrated Security=SSPI; MultipleActiveResultSets=true;"

location: C:\Users\Steven
test.mdf
test_log.ldf

## >Azure Scheduler Job

In Scheduler Job Collection, Add [name],Action settings[Https,Get,Url(below), Schedule Recurring setting]

https://abilityfirst-co.azurewebsites.net/executetask/?name=AddPostedJobsOver4HoursInQueue

```
<appSettings>
        <add key="AzureQueueName_Emails" value="emails"/>
        <add key="AzureQueueName_Jobs" value="jobs"/>
        <add key="AzureQueueName_PostedJobsOver4Hours" value="postedjobsover4hours" />
</appSettings>
```

C#, add AzureQueveService

```
public class AzureQueueService : IAzureQueueService
{
	#region Fields

	private readonly IWriteEntities _entities;
	private readonly CloudQueue _queue;

	#endregion

	#region Ctor

	public AzureQueueService(IWriteEntities entities)
	{
		this._entities = entities;
	}

	public AzureQueueService(IWriteEntities entities, CloudQueue queue)
	{
		this._entities = entities;
		this._queue = queue;
	}

	#endregion

	#region service

	public CloudQueueMessage GetQueueMessage(string queueName)
	{
		CloudQueue queue = GetQueue(queueName);
		CloudQueueMessage retrievedMessage = queue.GetMessage();
		if (retrievedMessage == null) return null;
		return retrievedMessage;
	}

	public void AddMessageInQueue(string queueName, string message)
	{
		CloudQueue queue = GetQueue(queueName);
		CloudQueueMessage queueMessage = new CloudQueueMessage(message);
		queue.AddMessage(queueMessage);
		//queue.AddMessage(queueMessage, TimeSpan.FromSeconds(30));
	}

	public string PeekMessageFromQueue(string queueName)
	{
		CloudQueue queue = GetQueue(queueName);
		CloudQueueMessage peekedMessage = queue.PeekMessage();
		return peekedMessage.AsString;
```

```
        }

        public void UpdateMessageInQueue(string queueName, string newMessage)
        {
                CloudQueue queue = GetQueue(queueName);
                CloudQueueMessage message = queue.GetMessage();
                message.SetMessageContent(newMessage);
                // Make it invisible for another 60 seconds.
                queue.UpdateMessage(message, TimeSpan.FromSeconds(60.0), MessageUpdateFields.Content |
MessageUpdateFields.Visibility);
        }

        public void DeleteMessageInQueueWithID(string queueName, string messageID, string messagePopReceipt)
        {
                CloudQueue queue = GetQueue(queueName);
                queue.DeleteMessage(messageID, messagePopReceipt);
        }

        public void DeleteMessageInQueue(string queueName, CloudQueueMessage queueMessage)
        {
                CloudQueue queue = GetQueue(queueName);
                //Process the message in less than 30(default) seconds, and then delete the message
                if (queueMessage != null)
                        queue.DeleteMessage(queueMessage);
        }

        public int GetQueueMessageCount(string queueName)
        {
                CloudQueue queue = GetQueue(queueName);
                queue.FetchAttributes();
                return (int)queue.ApproximateMessageCount;
        }

        public List<string> ProcessMessages(string queueName, int messageCount, int holdTime)
        {
                List<string> messageList = new List<string>();
                CloudQueue queue = GetQueue(queueName);
                foreach (CloudQueueMessage message in queue.GetMessages(messageCount,
TimeSpan.FromMinutes(holdTime)))
                {
                        // Process all messages in less than 5 minutes,
                        messageList.Add(message.AsString);

                        // Deleting each message after processing.
                        queue.DeleteMessage(message);
                }
                return messageList;
        }
```

```
        public List<CloudQueueMessage> GetMessagesFromQueue(string queueName, int numberOfMessages, int
holdTime)
        {
                List<CloudQueueMessage> messageList = new List<CloudQueueMessage>();
                CloudQueue queue = GetQueue(queueName);
                foreach (CloudQueueMessage message in queue.GetMessages(numberOfMessages,
TimeSpan.FromMinutes(holdTime)))
                {
                        // Process 'numberOfMessages' messages in less than 'holdTime' minutes
                        messageList.Add(message);
                }
                return messageList;
        }

        #endregion

        #region Helper

        private CloudQueue GetQueue(string queueName)
        {
                string connectionString = CloudConfigurationManager.GetSetting("StorageConnectionString");
                ////string connectionString = "UseDevelopmentStorage=true";   //this for testing only;

                CloudStorageAccount storageAccount = CloudStorageAccount.Parse(connectionString);
                CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();
                CloudQueue queue = queueClient.GetQueueReference(queueName);
                queue.CreateIfNotExists();
                return queue;
        }

        #endregion
}


In controler
//this part need to change later for id Authentication
[AllowAnonymous]
[HttpPost, Route("executetask")]
public ActionResult ExecuteTask(string name)
{
        switch (name)
        {
                case "AddPostedJobsOver4HoursInQueue":
                        int jobCount = PostedJobsOver4HoursProcess();
                        if (jobCount != 0)
                        {
                                var subject = "There have " + jobCount + " jobs posted over 4 hour still not filled !";
                                //          this._emailService.SendEmailToAllCoordinators(subject);
                        }
                        break;
```

```
                case "ClearUpQueueForPostedJobsOver4Hours":
                        DeleteJobsInQueueWhenStatusIsNotUrgent();
                        break;
                case "BookingAlertFor24Hours":
                        // booking service before arrive start 24 hours
                        break;
                case "BookingAlertFor1Hour":
                        // booking service before arrive 1 hours
                        break;
                case "sendEmail":
                        // testing for send alert email to coordinator when error, change later for admin
                        var note = "The job scheduler service has error";
                        //          this._emailService.SendEmailToCoordinator(3, note);
                        break;
                default:
                        break;
        }
        return new EmptyResult();
}

#endregion

#region Helper

private int PostedJobsOver4HoursProcess()
{
        string queueName = ConfigurationManager.AppSettings["AzureQueueName_PostedJobsOver4Hours"];
        int jobCounter = 0;
        DateTime urgentTime = DateTime.Now.AddHours(-jobPostedHours);

        /*
        var jobs = this._entities.Get<Job>(j => j.CreatedAt < urgentTime && (j.Status == JobStatus.New || j.Status ==
JobStatus.Updated));

        foreach (var job in jobs)
        {
                if (job.ServiceAt < DateTime.Now)
                {
                        job.Close();
                        // send email to client
                        // var subject = "The job scheduler service has error";
                        // this._notificationService.SendEmailToClient(clientID, subject);
                }
                else
                {
                        this._queueService.AddMessageInQueue(queueName, job.ID.ToString());
                        job.Urgent();
                        jobCounter++;
                }
                this._entities.Update(job);
```

```
            }
            */
            this._entities.Save();
            return jobCounter;
}

private void DeleteJobsInQueueWhenStatusIsNotUrgent()
{
            string queueName = ConfigurationManager.AppSettings["AzureQueueName_PostedJobsOver4Hours"];
            List<CloudQueueMessage> messages = this._queueService.GetMessagesFromQueue(queueName,
messageNumber, minute);
            foreach (var message in messages)
            {
                        int jobID = Convert.ToInt32(message.AsString);
                        Job retrievedJob = this._entities.Single<Job>(j => j.ID == jobID);
                        /*
                        if (retrievedJob != null && retrievedJob.Status != JobStatus.Urgent)
                                    this._queueService.DeleteMessageInQueue(queueName, message);
             */
            }
}

#endregion
```

>Deploy on Azure error:

1. Could not load file or assembly 'Google.Apis.Auth'

  >>Delete all .dll, or delete all files

2. loading X.509 certificates cannot find

  a. Azure upload and using certification
  https://azure.microsoft.com/en-us/blog/using-certificates-in-azure-websites-applications/

  ::does not support free version

```
using System.Security.Cryptography.X509Certificates;namespace UseCertificateInAzureWebsiteApp
{
 class Program
 {
  static void Main(string[] args)
  {
   X509Store certStore = new X509Store(StoreName.My, StoreLocation.CurrentUser);
   certStore.Open(OpenFlags.ReadOnly);
   X509Certificate2Collection certCollection = certStore.Certificates.Find(
                        X509FindType.FindByThumbprint,
                        // Replace below with your cert's thumbprint
```

```
                    "E661583E8FABEF4C0BEF694CBC41C28FB81CD870",
                    false);
    // Get the first cert with the thumbprint
    if (certCollection.Count > 0)
    {
      X509Certificate2 cert = certCollection[0];
      // Use certificate
      Console.WriteLine(cert.FriendlyName);
    }
    certStore.Close();
   }
  }
}
```

b. Loading the Certificate from Resource as byte[]
http://developers.de/blogs/armin_kalajdzija/archive/2015/08/14/how-to-use-client-certificate-in-azure-web-app.aspx

>Google analytics Auth2 get service with .p12, .json, and key
https://stackoverflow.com/questions/43365189/baseclientservice-initializer-not-found-in-google-api-in-c-sharp/43376166

>Add users in Google analytics

https://analytics.google.com/

left menu, to to Admin,
below the Accout, select user management
add email which ganerate by add service accounts
such as: myabilityfirst-1487808941446@appspot.gserviceaccount.com

>Get google analytics profile ID:

https://analytics.google.com/analytics/web/#report/trafficsources-all-traffic/a107999270w161251581p162383562/

profile id=162383562

>X509Certificate2 in local

Turns out there's a setting in the IIS Application Pool configuration (Application Pools > Advanced Settings) Under process model, find the Load User profile, set option as True

```
var keyFilePath = @"C:\Steven_Doc\angry-chicken2\src\MyAbilityFirst\App_Data\MyAbilityFirst-GoogleKey.p12";
 //loading the Key file
var certificate = new X509Certificate2(keyFilePath, "notasecret",
X509KeyStorageFlags.MachineKeySet);
```

>X509Certificate2 on Azure

```
var collection = new X509Certificate2Collection();
collection.Import(certPath, certPassPhrase, X509KeyStorageFlags.MachineKeySet);

foreach (var cert in collection)
{
   // Import the certificate into an X509Store object
   if (cert.Subject.Contains(subjectMatch))
   {
      return cert;
   }
}
```

https://stackoverflow.com/questions/31857930/the-system-cannot-find-the-file-specified-using-x509certificate2-to-connect-goog

>c# datetime format

| Format | E.g. Result |
|---|---|
| DateTime.Now.ToString("MM/dd/yyyy") | 05/29/2015 |
| DateTime.Now.ToString("dddd, dd MMMM yyyy") | Friday, 29 May 2015 |
| DateTime.Now.ToString("dddd, dd MMMM yyyy") | Friday, 29 May 2015 05:50 |
| DateTime.Now.ToString("dddd, dd MMMM yyyy") | Friday, 29 May 2015 05:50 AM |
| DateTime.Now.ToString("dddd, dd MMMM yyyy") | Friday, 29 May 2015 5:50 |
| DateTime.Now.ToString("dddd, dd MMMM yyyy") | Friday, 29 May 2015 5:50 AM |
| DateTime.Now.ToString("dddd, dd MMMM yyyy HH:mm:ss") | Friday, 29 May 2015 05:50:06 |
| DateTime.Now.ToString("MM/dd/yyyy HH:mm") | 05/29/2015 05:50 |
| DateTime.Now.ToString("MM/dd/yyyy hh:mm tt") | 05/29/2015 05:50 AM |
| DateTime.Now.ToString("MM/dd/yyyy H:mm") | 05/29/2015 5:50 |
| DateTime.Now.ToString("MM/dd/yyyy h:mm tt") | 05/29/2015 5:50 AM |
| DateTime.Now.ToString("MM/dd/yyyy HH:mm:ss") | 05/29/2015 05:50:06 |
| DateTime.Now.ToString("MMMM dd") | May 29 |
| DateTime.Now.ToString("yyyy'-'MM'-'dd'T'HH':'mm':'ss.fffffffK") | 2015-05-16T05:50:06.7199222-04:00 |
| DateTime.Now.ToString("ddd, dd MMM yyy HH':'mm':'ss 'GMT'") | Fri, 16 May 2015 05:50:06 GMT |
| DateTime.Now.ToString("yyyy'-'MM'-'dd'T'HH':'mm':'ss") | 2015-05-16T05:50:06 |
| DateTime.Now.ToString("HH:mm") | 05:50 |
| DateTime.Now.ToString("hh:mm tt") | 05:50 AM |
| DateTime.Now.ToString("H:mm") | 5:50 |
| DateTime.Now.ToString("h:mm tt") | 5:50 AM |
| DateTime.Now.ToString("HH:mm:ss") | 05:50:06 |
| DateTime.Now.ToString("yyyy MMMM") | 2015 May |

>C# csp disable in controller

```
[Authorize(Roles = "Client")]
[Csp(Enabled = false)]
public class JobController : Controller
{
}
```

>json get results

```
public enum HttpStatusCode
 {
   Continue = 100,
   SwitchingProtocols = 101,
   OK = 200,
   Created = 201,
   Accepted = 202,
   NonAuthoritativeInformation = 203,
   NoContent = 204,
   ResetContent = 205,
   PartialContent = 206,
   Ambiguous = 300,
   MultipleChoices = 300,
   Moved = 301,
   MovedPermanently = 301,
   Found = 302,
   Redirect = 302,
   RedirectMethod = 303,
   SeeOther = 303,
   NotModified = 304,
   UseProxy = 305,
   Unused = 306,
   RedirectKeepVerb = 307,
   TemporaryRedirect = 307,
   BadRequest = 400,
   Unauthorized = 401,
   PaymentRequired = 402,
   Forbidden = 403,
   NotFound = 404,
   MethodNotAllowed = 405,
   NotAcceptable = 406,
   ProxyAuthenticationRequired = 407,
   RequestTimeout = 408,
   Conflict = 409,
   Gone = 410,
   LengthRequired = 411,
   PreconditionFailed = 412,
   RequestEntityTooLarge = 413,
   RequestUriTooLong = 414,
   UnsupportedMediaType = 415,
```

```
  RequestedRangeNotSatisfiable = 416,
  ExpectationFailed = 417,
  InternalServerError = 500,
  NotImplemented = 501,
  BadGateway = 502,
  ServiceUnavailable = 503,
  GatewayTimeout = 504,
  HttpVersionNotSupported = 505,
 }
```

>Azure debug

```
  https://<sitename>.scm.azurewebsites.net
```

>JS reload page 9/26/2017

```
  location.replace(location.pathname);
  window.location.reload();
```

>EF update error fix:

error 1. The operation failed: The relationship could not be changed because one or more of the foreign-key properties is non-nullable
>> fix using: this._entities.Delete(schedule);
error 2. Collection was modified; enumeration operation may not execute
>> fix using: Schedule schedule in job.Schedules.ToList();

```
//delete old list
foreach (Schedule schedule in job.Schedules.ToList())
        this._entities.Delete(schedule);
//clear new in list
job.ClearSchedules();
//add new item in list
foreach (var schedule in getJobSchedules(model))
        job.AddSchedule(schedule);
```

>localStorage vs sessionStorage

localStorage must be deleted menually
sessionStorage base on current page, window is closed, it is deleted.

>JS localStorage

```
//Convert array to json and save in localStorage:
  var events = [];
        ...
  localStorage.setItem('ExitingSchedules', JSON.stringify(events));
```

```
//Read localStorge and convert to array:

	function loadExitingSchedules() {
		var stored = localStorage.getItem('ExitingSchedules');
		if (stored != null) {
			var result = JSON.parse(stored);
			var arr = [];
			for (var i in result)
				arr.push(result[i]);
			return arr;
		}
		else {
			getExitingSchedules().OneYearDuration;
			alert("Network issues, please try again! ");
			return null;
		}
	};

//Remove item
localStorage.removeItem('ExitingSchedules');

//clear
window.localStorage.clear();

//clear localStorage
$(function () {
	window.clearLocalStorage = function () {
		window.localStorage.clear();
		location.reload();
		return false;
	};
});


>fullcalendar add two eventsource from json

following code work but -- Uncaught TypeError: Cannot read property 'clone' of undefined
-------- method one -----------------

var exitingSchedules = {
	OneYearDuration: {
		events: function (start, end, timezone, callback) {
		$.ajax({
			url: $('#calendar-url').val(),
			cache: true,
			type: 'GET',
			id: existingScheduleID,
			title: "exsiting event",
			dataType: 'json',
```

```
                        data: { jobID: jobID },
                        success: function (result) {
                                var events = [];
                                $.each(result, function (i, item) {
                                var event = {
                                        id: result[i].ID.toString(),
                                        title: result[i].Title,
                                        start: moment(result[i].StartTime, "YYYY-MM-DD HH:mm").format(),
                                        end: moment(result[i].EndTime, "YYYY-MM-DD HH:mm").format(),
                                        allDay: false,
                                        editable: false,
                                        //          overlap: true,
                                        className: 'requests',
                                        color: result[i].Color
                                };
                                events.push(event);
                                });
                                callback && callback(events);
                        },
                        error: function () {
                                alert('There was an error while fetching events!');
                        }
                })}
        }
};


if (existingEventsSource === null)
        existingEventsSource = exitingSchedules.OneYearDuration;   /// <<<<<<<
 $('#calendar').fullCalendar('addEventSource', existingEventsSource);
```

the following code working but --  Uncaught TypeError: Cannot read property 'format' of undefined

-------- method two ----------

```
var exitingSchedules = {
        OneYearDuration: {
                url: $('#calendar-url').val(),
                …
                success: function (result) {
                        var events = [];
                        $.each(result, function (i, item) {
                                var event = {
                                        …
                                };

                                events.push(event);
                        });
                         return events;
                },
```

```
                error: function () {
                        alert('There was an error while fetching events!');
                }

        }
};

-------- method three ----------------
function loadExitingSchedules() {
                if (existingEventsSource === null)
                        existingEventsSource = getExitingSchedules().OneYearDuration;    ///<<<< need
                $('#calendar').fullCalendar('addEventSource', existingEventsSource);
        };

function getExitingSchedules() {
        overLapWarning = true;
        var jobID = $('#ID').val() != null ? $('#ID').val() : 0;
        var exitingSchedules = {
                OneYearDuration: {
                        url: $('#calendar-url').val(),
                        …
                        success: function (result) {
                                var events = [];
                                $.each(result, function (i, item) {
                                        var event = {
                                        ……
                                        };
                                        events.push(event);
                                });
                                return events;  //<<<<<<<<<<<<<< need
                        },
                        error: function () {
                                alert('There was an error while fetching events!');
                        }
                }
        };
        return exitingSchedules;  //<<<<<<<<<<<<<< need
}

solution: addEventSource delay 300sec, make sure calendar UI has open
if ($('#existingjob-switch').prop('checked')) {
        window.setTimeout(loadExitingSchedules, 300);

        function loadExitingSchedules() {
                if (existingEventsSource === null)
                        existingEventsSource = exitingSchedules.OneYearDuration
                $('#calendar').fullCalendar('addEventSource', existingEventsSource);
        }
}
```

>Js call function in other JS

```
window.getMaxEndDate=function (inputedDateTime) {
        var duration = moment.duration({ 'year': 1 });
        var endTime = moment(inputedDateTime).add(duration);
        var maxTime = moment(endTime);
        return maxTime;
}
```

```
var maxdate=getMaxEndDate(date);
```

>C# ISO 8601 Extension

```
string isoDate = DateTime.UtcNow.ToIso8601Date();
DateTime date = isoDate.FromIso8601Date();

/// <summary> /// DateTime Iso Extensions. /// </summary>
public static class DateTimeIsoExtensions
{
    private const string DateTimeFormat = "{0}-{1}-{2}T{3}:{4}:{5}Z";

    /// <summary> /// To the iso8601 date. /// </summary>
    ///The date.
    /// <returns></returns>
    public static string ToIso8601Date(this DateTime date)
    {
        return string.Format(
            DateTimeFormat,
            date.Year,
            PadLeft(date.Month),
            PadLeft(date.Day),
            PadLeft(date.Hour),
            PadLeft(date.Minute),
            PadLeft(date.Second));
    }

    /// <summary> /// Froms the iso8601 date. /// </summary>
    ///The date.
    /// <returns></returns>
    public static DateTime FromIso8601Date(this string date)
    {
        return DateTime.ParseExact(date.Replace("T", " "), "u",
CultureInfo.InvariantCulture);
    }

    private static string PadLeft(int number)
    {
        if (number < 10)
        {
            return string.Format("0{0}", number);
        }

        return number.ToString(CultureInfo.InvariantCulture);
```

```
    }
}
```

>Fullcalendar  Event datetime:

https://en.wikipedia.org/wiki/ISO_8601

```
        dateString = d.Start.ToString("o");    //datetime to ISO C#

        StartTime = String.Format("{0:yyyy-MM-dd HH:mm:ss tt}", d.Start),
        moment(dateString, "YYYY-MM-DD HH:mm").format();// js for fullcalendar : "2017-09-25T13:35:00+10:00"

        String.Format("{0:u}", d.Start),  Z
```

```
String.Format("{0:dd/MM/yyyy HH:mm:ss}", dt);    //  09/03/2008 16:05:07" - english (en-US)
string endDate = String.Format("{0:yyyy-MM-dd HH:mm:ss tt}", d.End),
```

Js:

```
        start: moment(result[i].Start, "YYYY-MM-DD HH:mm"),
          end: moment(result[i].End, "YYYY-MM-DD HH:mm"),
```

>Random color

```
Random randomGen = new Random();
private string getRandomColor()
{
        KnownColor[] names = (KnownColor[])Enum.GetValues(typeof(KnownColor));
        KnownColor randomColorName = names[randomGen.Next(names.Length)];
        var color = Color.FromKnownColor(randomColorName);
        var colorString=  $"#{color.R:X2}{color.G:X2}{color.B:X2}";
        return colorString;
}
```

```
Random rand = new Random();
string color = getRandomColor(rand);
private string getRandomColor(Random rand)
{
        Color color = Color.FromArgb(rand.Next(256), rand.Next(256), rand.Next(256));
        string colorString=         $"#{color.R:X2}{color.G:X2}{color.B:X2}";
        return colorString;
}
```

```
var color = Color.FromArgb(rand.Next(256) & 0x7F7F7F, rand.Next(256) & 0x7F7F7F, rand.Next(256) & 0x7F7F7F);
//dark color
var color= Color.FromArgb(random.Next(200, 255), random.Next(150, 255), random.Next(150, 255));  //light color
```

if you want lighter color you can choose a lighter html color  code
System.Drawing.ColorTranslator.FromHtml("#ColorCode");

https://www.w3schools.com/colors/colors_picker.asp

>Js Confirm dialog 9/20/2017

```
<input type="submit" value="Save" id="submitcheck" class="btn btn-default btn-sm" />

$('#submitcheck').click(function () {
    return confirm("Confirm delete?");
}
```

>Compare two date C#

```
@if (DateTime.Compare(item.EndDate, DateTime.Now)>0)
{}
```

>EF error (Automap, AutoFac) Solution:

The operation failed: The relationship could not be changed because one or more of the foreign-key properties is non-nullable.

Delete operation:

```
client.DeleteJob(jobID);
this._entities.Delete(job);      //delete children first
this._entities.Update(client);
this._entities.Save();
```

Update operation:

```
var job = this._entities.Get<Job>(j => j.ID == model.JobID).FirstOrDefault();      //get old data first
job = _mapper.Map<NewJobViewModel, Job>(model, job);                               //map model to old data
foreach (var patientID in model.PatientIDs)
{
        var patient = this._entities.Single<Patient>(c => c.ID == patientID);
        if (model.AddressSameProfile)
                job.Address = patient.Address;
        job.AddPatient(patient);
}
foreach (var element in getJobSchedule(model))
job.AddSchedule(element);

if (job.Schedules != null && job.Patients != null)
{
        var updateJob = client.UpdateJob(job);
        if (updateJob != null)
        {
                this._entities.Update(client);
```

```
                this._entities.Save();
        }
}
```

>Convert string to list

convert string to string list:

```
var foos = "Foo1,Foo2,Foo3";
var fooArray = foos.Split(',');
```

convert string to int list:

```
var selectedJobsID = model.SelectedJobsID.Split(',').Select(int.Parse).ToList();
```

>Send array to controller

9/19/2017

```
public string SelectedJobsID { get; set; }

@Html.HiddenFor(model => model.FirstOrDefault().SelectedJobsID)
@{ var j = 0; }
<div class="delete-container">
@foreach (var item in Model)
{
        @Html.CheckBoxFor(modelItem => item.SelectedJobs[j], new { @style = "cursor:pointer", @id = "selected"
+ item.ID })
        j++;
}
</div>


<script>
        $(function () {
                $('.delete-container').change(function () {
                        var jobsid = [];
                        $('.delete-container input[type=checkbox]').each(function () {
                                if (this.checked) {
                                        var id = this.id.substr('selected'.length);
                                        jobsid.push(id);
                                }
                        });
                        $('#SelectedJobsID').val(jobsid);
                        if (jobsid.length > 0)
                                $('#deleteBtn').show();
                        else
                                $('#deleteBtn').hide();
                })
```

```
        });
</script>

Post:
var selectedJobsID = model.SelectedJobsID.Split(',').Select(int.Parse).ToList();
```

>JS function

```
$(document).ready(function () {

 jQuery(function ($) {

 $(function () {
```

>TempData post to different controller

```
1:
[HttpPost, Route("jobs")]
public ActionResult Jobs(JobsViewModel model)
{
        if (ModelState.IsValid)
        {
                TempData["model"] = model;
                return RedirectToAction("DeleteJob");
        }
        return RedirectToAction("jobs");
}

[HttpGet]
 public ActionResult DeleteJob()
{
 JobsViewModel model = (JobsViewModel)TempData["model"];
 return View(model);
}

or
@using (Html.BeginForm("delete-job", "Job", FormMethod.Post, new { enctype = "multipart/form-data" }))
{
 <input type="submit" value="Delete Selected Job"  class="btn btn-warning btn-xs" style="display: none;"
id="deleteBtn"/>
}

[HttpPost, Route("job/delete-job")]
public ActionResult DeleteJob(JobsViewModel model)
{
        var client = this.GetLoggedInUser();
        if (ModelState.IsValid)
                this._clientServices.DeleteJobs(client.ID, model);
        return RedirectToAction("jobs");
}
```

2: using Query string data passed

Or you can frame it with the help of query strings like

return RedirectToAction("GetStudent","Student", new {Name="John", Class="clsz"});
This will generate a GET Request like

Student/GetStudent?Name=John & Class=clsz

Update and delete
The operation failed: The relationship could not be changed because one or more of the foreign-key properties is non-nullable.

Mapper.CreateMap<MyDataContract, MyEntity>
ForMember(dest => dest.NavigationProperty1, opt => opt.Ignore())
ForMember(dest => dest.NavigationProperty2, opt => opt.Ignore())

> Model List to JS Array

```
var p = JSON.parse('@Html.Raw(Json.Encode(@Model.SelectedPatients))');
$('#patients-container label').each(function () {
if (p[i-1]==true)  //skip first label
{
       var selected = $(this).children("input[type=checkbox]");
       $('#'+selected.id).prop('checked', true);
       $(this).addClass("active");
       }
       i++;
})
```

>HashSet vs List

- List<T> when you want to:  Store a collection of items in a certain order.
- Hashset<T> when you want to:  Quickly find out if a certain object is contained in a collection.
    - A HashSet is a List with no duplicate members.

>EF Configure Many-to-Many relationship:
http://www.entityframeworktutorial.net/code-first/configure-many-to-many-relationship-in-code-first.aspx

```
public class Student
{
   public int StudentId { get; set; }
```

```
    public string StudentName { get; set; }
    public virtual ICollection<Course> Courses { get; set; }

    public Student()
    {
        this.Courses = new HashSet<Course>();   //or List
    }
}

public class Course
{
    public int CourseId { get; set; }
    public string CourseName { get; set; }
    public virtual ICollection<Student> Students { get; set; }

    public Course()
    {
        this.Students = new HashSet<Student>();   //or List
    }
}
```

>JS get dropdownlist selected text

```
        var recurringfrequency = $('#frequency-selecting').val();
        var selectedfrequency = $('#frequency-selecting').children("option:selected").text();
        var sss = $('#frequency-selecting').find("option:selected").text();
```

>JS simulate press key

```
$('#full-address').click();
setTimeout(simulatepressentry, 300);

function simulatepressentry() {
        $('#full-address').focus();
        var e = jQuery.Event("keydown");
        e.which = 13;  //  13-entry, 40-downarrow key
        e.keyCode = 13;
        $('#full-address').trigger(e);
}
```

>c# get and remove the last item in array

```
 var lastDate = dates[dates.Count() - 1];   //get last item
 var newDates= dates.Take(dates.Length-1);  //remove last item
```

>JS focus and click event

```
document.getElementById('patientinfo').focus();
document.getElementById('patientinfo').click();
```

>JS set array value to model mvc

```
public List<string> EventsDate { get; set; }

var eventsdatearray =[];
while (allDates[day]) {
	var eventDate = moment(allDates[day]).format("YYYY-MM-DD");
	eventsdatearray.push(eventDate);
	.....
}
$('#EventsDate').val(eventsdatearray);

@Html.HiddenFor(model => model.EventsDate)
```

>menully validation for muti checkbox

```
<!-- begin week selecting -->
<span>   <b>On</b></span>
<div class="btn-group" data-toggle="buttons" id="dayofweek-container">
	@{
		var j = 0;
		foreach (var element in Model.DayofWeekDropdownList)
		{
		<label class="btn btn-default button_day" style="padding:2px 4px 2px 4px;min-width:30px">
			@Html.CheckBoxFor(model => model.SelectedDayofWeek[j], new { @id = "dayofweek" + j,
@title = @element.Text }) <span> @element.Text.Substring(0, 3)</span>

		</label>
			j++;
		}
	}
	<br />
</div>
@Html.ValidationMessageFor(model => model.SelectedDayofWeek, "", new { @class = "text-danger", @id =
"dayofweekvalidation" })
<!-- end week selecting -->



$('#submitcheck').click(function () {
	if ($('#recurring-switch').prop('checked') && $('#frequency-selecting').val() == "Weekly") {
		var selected = false;
		$('#dayofweek-container').find('input:checked').each(function () {
			var ss = this;
			selected = true;
		});
		if (selected)
			document.createElement('form').submit.call(document.getElementById('jobform'));
		else {
```

```
                $('#dayofweekvalidation').html("Please select at least one option !");
                return false;
            }
        }
});
```

>Validation for hide field and multiple dropdownlist

::> style="visibility:hidden;"

```
<div class="col-md-12" id="patients-container">
        @Html.LabelFor(model => model.PatientId, new { @class = "job-label" })<br />
        @{
            var i = 0;
            foreach (var element in Model.PatientDropDownList)
            {
            <div class="btn-group" data-toggle="buttons" style="padding:3px;">
                <label class="btn btn-default button_patientname">
                    @Html.CheckBoxFor(model => model.SelectedPatients[i], new { @id = "patientid"
+ element.Value }) <span> @element.Text</span>
                </label>
            </div>
                i++;
            }
        }

        @Html.DropDownListFor(model => model.PatientIDs, (MultiSelectList)Model.PatientDropDownList, new
{ @class = "form-control", style = "height:0px;padding:0px; margin:0;visibility:hidden;", multiple = "multiple" })
        @Html.ValidationMessageFor(model => model.PatientIDs, "", new { @class = "text-danger"})
</div>
```

>Example to import ICS file into fullcalendar event

```
include : jquery.ics-0.3.js (google is your friend)

in your js file :

$.ajax({
            type: 'POST',
            contentType: "application/json",
            url: *****ics file*****,
            dataType: "json",
            async: false,
            success: function(result) {
                var cal, icalevents, dtstart, dtend, StrStart, StrEnd, StrID, StrAllDay,
StrTitle, StrURL, StrLocation, StrDescription;
                $.each(result, function(key, val) {
                    try {
                        cal = $.parseIcs(val.url, AEFC.urls.externalfile);
```

```
                        Alertify.log.success('ICS '+val.url);
                        icalevents = [];

                        for (i = 0; i < cal.event.length; i = i + 1) {
                            dtstart = cal.event[i].dtstart[0].value;
                            StrStart = dtstart.substring(0, 4) + '-' + dtstart.substring(4,
6) + '-' + dtstart.substring(6, 8);
                            if (dtstart.substring(9, 11) !== '') {
                                StrStart = StrStart + 'T' + dtstart.substring(9, 11)
+ '-' + dtstart.substring(11, 13) + '-' + dtstart.substring(13, 15);
                            }

                            dtend = cal.event[i].dtend[0].value;
                            StrEnd = dtend.substring(0, 4) + '-' + dtend.substring(4,
6) + '-' + dtend.substring(6, 8);
                            if (dtend.substring(9, 11) !== '') {
                                StrEnd = StrEnd + 'T' + dtend.substring(9, 11) + '-'
+ dtend.substring(11, 13) + '-' + dtend.substring(13, 15);
                            }

                            StrAllDay = StrStart.indexOf("T") === -1;

                            if (cal.event[i].uid !== "undefined") { StrID =
'ICS'+AEFC.nbcalendars+'#'+cal.event[i].uid[0].value
;}
                            if (cal.event[i].summary !== "undefined") { StrTitle =
cal.event[i].summary[0].value;}

                            //if (cal.event[i].url !== "undefined") { StrURL =
cal.event[i].url[0].value;}

                            //if (cal.event[i].location!== "undefined") { StrLocation =
cal.event[i].location[0].value;}

                            //if (cal.event[i].description !== "undefined") { StrDescription =
cal.event[i].description[0].value;}


                            icalevents.push({
                                id: StrID,
                                title: StrTitle,
                                url: StrURL,
                                start: StrStart,
                                end: StrEnd,
                                allDay: StrAllDay,
                                location: StrLocation,
                                description: StrDescription,
                                editable:false,
                                className: val.className,
                                color: val.color,
                                textColor: val.textColor
                            });
                        }
                        AEFC.source[AEFC.nbcalendars] = {
                            events: icalevents
                            //className: val.className,
                            //color: val.color,
                            //textColor: val.textColor
                        };
                    } catch (err) {
                        Alertify.log.error("ERROR loading (" + val.url + "): "+err);
```

```
                    }
                });
            }
        }); // $.ajax(
```

>Caculate recurring by every week and occurrence

```
if ($("#enddate-after").is(":checked")) {
        occurrence = $('#occurrence-weekly').val();
        var weeknumber = parseInt(occurrence);
        weeknumber = weeknumber * parseInt(sequenceweekly);
        var saturday = moment(startDateTime).day("Saturday");
        var duration = moment.duration({ 'week': parseInt(weeknumber) });
        endDateTime = moment(saturday).add(duration);
        duration = moment.duration({ 'days': moment(startDateTime).day() - 7 });
        endDateTime = moment(endDateTime).add(duration);

}

>>>>    every * occurrence + 1 ==week
```

| startdate | every | occurrence | enddate | week |
|---|---|---|---|---|
| 25-Aug | 1 | 1 | 31-Aug | 2 |
| 25-Aug | 1 | 2 | 7-Sep | 3 |
| 25-Aug | 1 | 3 | 14-Sep | 4 |
| 25-Aug | 1 | 4 | 21-Sep | 5 |
| 25-Aug | 1 | 5 | 28-Sep | 6 |
| 25-Aug | 2 | 1 | 7-Sep | 3 |
| 25-Aug | 2 | 2 | 21-Sep | 5 |
| 25-Aug | 2 | 3 | 5-Oct | 7 |
| 25-Aug | 2 | 4 | 19-Oct | 9 |
| 25-Aug | 2 | 5 | 2-Nov | 11 |
| 25-Aug | 3 | 1 | 14-Sep | 4 |
| 25-Aug | 3 | 2 | 5-Oct | 7 |
| 25-Aug | 3 | 3 | 26-Oct | 10 |
| 25-Aug | 3 | 4 | 16-Nov | 13 |
| 25-Aug | 3 | 5 | 7-Dec | 16 |
| 25-Aug | 4 | 1 | 21-Sep | 5 |
| 25-Aug | 4 | 2 | 19-Oct | 9 |
| 25-Aug | 4 | 3 | 16-Nov | 13 |
| 25-Aug | 4 | 4 | 14-Dec | 17 |
| 25-Aug | 5 | 1 | 28-Sep | 6 |
| 25-Aug | 5 | 2 | 2-Nov | 11 |
| 25-Aug | 5 | 3 | 7-Dec | 16 |

>Fullcallendar tips:

For recurring events dont use the addEventSource function,

>moment-recur

https://github.com/c-trimm/moment-recur

```
var myDate, recurrence;

// Create a date to start from
myDate = moment("01/01/2014");

// You can pass the units to recur on, and the measurement type.
recurrence = myDate.recur().every(1, "days");

// You can also chain the measurement type instead of passing it to every.
recurrence = myDate.recur().every(1).day();

// It is also possible to pass an array of units.
recurrence = myDate.recur().every([3, 5]).days();

// When using the dayOfWeek measurement, you can pass days names.
recurrence = myDate.recur().every(["Monday", "wed"]).daysOfWeek();

// Month names also work when using monthOfYear.
recurrence = myDate.recur().every(["Jan", "february"], "monthsOfYear");

<script src="~/Scripts/moment-recur.js"></script>

var myDate, interval;

// Create a date to start from
myDate = moment("01/01/2014");

// A daily interval - will match every day.
interval = myDate.recur().every(1).day();

// A bi-weekly interval - will match any date that is exactly 2 weeks from myDate.
interval = myDate.recur().every(2).weeks();

// A quarterly interval - will match any date that is exactly 3 months from myDate.
interval = myDate.recur().every(3).months();

// A yearly interval - will match any date that is exactly 1 year from myDate.
interval = myDate.recur().every(1).years();
```

```
// It is possible to match multiple units of a single measure using an array.
interval = myDate.recur().every([3, 5]).days();

// It is NOT possible to create compound intervals. The following will never match.
interval = myDate.recur().every(3).days().every(2).months(); // Won't work
```

Calendar Intervals

```
var cal;

// Will match any date that is on Sunday or Monday.
cal = moment.recur().every(["Sunday", 1]).daysOfWeek();

// Will match any date that is the first or tenth day of any month.
cal = moment.recur().every([1, 10]).daysOfMonth();

// Will match any date that is in the first or third week of any month.
cal = moment.recur().every([1, 3]).weeksOfMonth();

// Will match any date that is in the 20th week of any year.
cal = moment.recur().every(20).weekOfYear();

// Will match any date that is in January of any year.
cal = moment.recur().every("January").monthsOfYear();

// You can also combine these rules to match specific dates.
// For instance, this will match only on Valentines day
var valentines = moment.recur().every(14).daysOfMonth()
                       .every("Februray").monthsOfYear();

// A weekOfMonthByDay interval is available for combining with
// the daysOfWeek to achieve "nth weekday of month" recurrences.
// The following matches every 1st and 3rd Thursday of the month.
// (Note this cannot be combined at the moment with every(x).months() expression)
cal = moment.recur().every("Thursday").daysOfWeek()
                .every([0, 2]).weeksOfMonthByDay();

cal = moment.recur().every(moment("01/01/2014").day()).daysOfWeek()
                .every(moment("01/01/2014").monthWeekByDay()).weeksOfMonthByDay();
```

>Fullcanlendar add event

```
var events = [];

var event = new Object();
var event = {
            id: index,
            title: 'updated event1',
            start: startTime,
            end: endTime,
            dow: selecteddayofweeek,
```

```
	ranges: [{
		start: moment(startDateTime).startOf('week'), //next two weeks
		end: moment(startDateTime).endOf('week').add(7, 'd'),
	}, {
		start: moment(startDateTime, 'YYYY-MM-DD'), //all of february
		end: moment(startDateTime, 'YYYY-MM-DD').endOf('month'),
	}],
};

events.push(event);
$('#calendar').fullCalendar('addEventSource', events);
// or  $('#calendar').fullCalendar('renderEvent', event, true);   ///must be have title
```

>Multiple select DayOfWeek

```
public List<int> DayofWeeks { get; set; }
public List<bool> SelectedDayofWeek { get; set; }
public IEnumerable<SelectListItem> DayofWeekList
{
	get
	{
		return Enum.GetValues(typeof(DayOfWeek)).Cast<DayOfWeek>()
					.Select(a => new SelectListItem { Value = ((int)a).ToString(), Text =
a.ToString() }).ToList();
	}
}
//check selected day of week
$('#dayofweek-container').change(function () {
	var days = [];
	$('input[type=checkbox]').each(function () {
		if (this.checked) {
			var id = this.id.substr('dayofweek'.length);
			days.push(id);
		}
	});
	$('#DayofWeeks').val(days);
})


<div class="btn-group" data-toggle="buttons" id="dayofweek-container">
	@{
		var j = 0;
		foreach (var element in Model.DayofWeekList)
		{
		<label class="btn btn-default button_day" style="padding:2px 4px 2px 4px;min-width:30px">
			@Html.CheckBoxFor(model => model.SelectedDayofWeek[j], new { @id = "dayofweek"
+ j }) <span> @element.Text.Substring(0, 3)</span>
		</label>
			j++;
		}
	}
	@Html.DropDownListFor(model => model.DayofWeeks, new MultiSelectList(Model.DayofWeekList,
"Value", "Text"), new { @class = "form-control", multiple = "multiple" })
```

```
</div>

>Convert DayOfWeek eum to SelectListItem

public IEnumerable<SelectListItem> DayofWeekSelectedList {
 get
 {
    return
    Enum.GetValues(typeof(DayOfWeek)).Cast<DayOfWeek>()
    .Select(a => new SelectListItem { Value = ((int)a).ToString(), Text = a.ToString() }).ToList();
 }
}

>Convert SelectListItems to SelectList
SelectList selectList = new SelectList(selectListItems, "Value", "Text");

>Convert SelectListItems to MultiSelectList
@Html.DropDownListFor(model => model.DayofWeeks, new MultiSelectList(Model.DayofWeekSelectedList,
"Value", "Text"), new { @class = "form-control", multiple = "multiple" })


>Enum to dropdownlist
(1)
@Html.DropDownListFor(model => model.DayofWeeks, EnumHelper.GetSelectList(typeof(DayOfWeek)), new
{ @class = "form-control"})

(2)
public DayOfWeek DayOfWeek { get; set; }
@Html.EnumDropDownListFor(model => model.DayOfWeek,new { @class = "form-control" })


var a=Enum.GetNames(typeof(DayOfWeek));
var b=Enum.GetValues(typeof(DayOfWeek)).OfType<DayOfWeek>().ToList();

var c=Enum.GetValues(typeof(DayOfWeek)).Cast<DayOfWeek>()
            .Select(dow => new { Value = (int)dow, Text = dow.ToString() })
            .ToList();



>Multiple Dropdownlist

 public int [] PatientIDs { get; set; }   //or   public List<int> PatientIDs { get; set; }
 public List<bool> SelectedPatients { get; set; }

<script>
//check selected patients
$('#patients-container').focusout(function () {
            var patients = [];
            var selectpatients = 0;
            $('input[type=checkbox]').each(function () {

                        if (this.checked) {
                                    var id = this.id.substr('Patientid'.length);
                                    selectpatients++;
```

```
                        patients.push(id);
                        $('#PatientId').val(id);
                }
        });

        $('#PatientIDs').val(patients);

        if (selectpatients == 1) {
                $('.same-patient-profile').show();
        }
        else {
                $('.same-patient-profile').hide();
                if ($('#address-selecting').prop('checked'))
                        $('#address-selecting').click();
        }

})
</script>
<div class="col-md-12" id="patients-container">
        @Html.LabelFor(model => model.PatientId, new { @class = "job-label" })<br />
        @{
                var i = 0;
                foreach (var element in Model.PatientDropDownList)
                {
                <div class="btn-group" data-toggle="buttons" style="padding:3px;">
                        <label class="btn btn-default button_patientname">
                                @Html.CheckBoxFor(model => model.SelectedPatients[i], new { @id =
"patientid" + element.Value }) <span> @element.Text</span>

                        </label>
                </div>
                        i++;
                }
        }

        @Html.DropDownListFor(model => model.PatientsID, (MultiSelectList)Model.PatientDropDownList, new
{ @class = "form-control", style = "display: none;", multiple = "multiple" })
        @Html.ValidationMessageFor(model => model.SelectedPatients, "", new { @class = "text-danger" })
</div>
```

>JS Pass array to controller

JavaScript

```
$("#getButtonValue").click(function (e) {
    e.preventDefault();
    var list = [];
    for (var i = 1; i < counter; i++) {
        list.push($('#textbox' + i).val());
    }
    var postData = { values: list };
    $.ajax({
        type: "POST",
        url: "/Surveys/PostQuestionAndOptions",
```

```
        data: postData,
        success: function (data) {
            alert(data);
        },
        dataType: "json",
        traditional: true
    });
});
```

Strongly typed object

```
public MyValues {
    public list<string> values {get; set;}
}
```

Controller method

```
[HttpPost]
public JsonResult PostQuestionAndOptions(MyValues model) {
    return Json(true, JsonRequestBehavior.AllowGet);
}
```

---

```
var orderModel = [];
//loop all the inputs in the page to get values, let's say you give all the inputs a class named'.orderinput'
$('#checkout').click(function(){
    $('.orderinput').each(function(){
    orderModel.push({Cylinder:$(this).val(),Sphere:blah,Quantity:blah,...);
    });
  });

//not all values are sotred into the orderModel, the do the post
$.ajax({
        url: 'your url',
        data:JSON.stringify(orderModel),
        type: 'POST',
        contentType: 'application/json; charset=utf-8',//this is important!!
        success : function(msg) {
            //blah..
        },
        error: function (xhr, ajaxOptions, thrownError) {
            //blah...
        }
    });
```

## >Js get model value

```
var jobID =  '@Model.ID';
```

## >Js pass model value to array

```
<script type="text/javascript">
    var myArray = [];
    @foreach (var d in Model.data)
    {
        @:myArray.push("@d");
```

```
    }
    alert(myArray);
</script>
```

>Get address from server 9/15/2017

```
function getPatientInfo(id) {
        var url = $('#request-url').val();
        $("#patientinfo").empty();
        $.ajax({
                type: 'GET',
                url: url,
                dataType: 'json',
                data: {
                        id: id
                },
                success: function (elements) {
                        $.each(elements, function (i, element) {
                                if ($('#address-selecting').prop('checked')) {
                                        document.getElementById(element.Text).value = element.Value;
                                }
                        });
                },
                error: function (ex) {
                        alert('Failed to retrieve data' + ex);
                }
        });
        return false;
};

public JsonResult GetPatientInfo(string id)
{
        var client = _clientServices.RetrieveClient(this.GetLoggedInUser().ID);
        var patient = this._clientServices.RetrievePatient(client.ID, int.Parse(id));
        DateTime now = DateTime.Today;
        DateTime dob = patient.DoB ?? DateTime.Now;
        int age = now.Year - dob.Year;
        if (now < dob.AddYears(age))
                age--;
        var interests = this._presentationService.GetSubCategoryListByUser("Interest", patient.ID);
        string tag = "";
        foreach (var element in interests)
        {
                tag = element.Name + ", " + tag;
        }

        List<SelectListItem> results = new List<SelectListItem>();
        results.Add(new SelectListItem { Text = "full-address", Value = patient.Address.FullAddress });
        results.Add(new SelectListItem { Text = "Address_Suburb", Value = patient.Address.Suburb });
        results.Add(new SelectListItem { Text = "Address_State", Value = patient.Address.State });
        results.Add(new SelectListItem { Text = "Address_Postcode", Value =
patient.Address.Postcode.ToString() });
        results.Add(new SelectListItem { Text = "Address_Latitude", Value =
patient.Address.Latitude.ToString() });
        results.Add(new SelectListItem { Text = "Address_Longitude", Value =
patient.Address.Longitude.ToString() });
```

```
        return Json(results, JsonRequestBehavior.AllowGet);
}
```

>EditorTemplates - for google places 08/16/17

---
~\Views\Shared\EditorTemplates\address.cshml

```
@model MyAbilityFirst.Domain.Address

<div class="geocomplete-container">
        @Html.TextBoxFor(
                model => model.FullAddress,
                new { @class = "form-control geocomplete-address", @placeholder = "street no. street, suburb,
state, postcode, country" ,@id="full-address", @required = true })
        @Html.ValidationMessageFor(model => model.FullAddress, "", new { @class = "text-danger" })
        <div class="geocomplete-details">
                @Html.HiddenFor(model => model.Suburb, new { @name = "profileSuburb", @data_geo =
"locality", @value = "" })
                @Html.HiddenFor(model => model.State, new { @name = "profileState", @data_geo =
"administrative_area_level_1", @value = "" })
                @Html.HiddenFor(model => model.Postcode, new { @name = "profilePostcode", @data_geo =
"postal_code", @value = "" })
                @Html.HiddenFor(model => model.Latitude, new { @name = "profileLatitude", @data_geo =
"lat", @value = "" })
                @Html.HiddenFor(model => model.Longitude, new { @name = "profileLongitude", @data_geo =
"lng", @value = "" })
        </div>
</div>
```

---
```
public class Address
{
        public string Suburb { get; set; }
        public string State { get; set; }
        public int Postcode { get; set; }
        public string FullAddress { get; set; }
        public decimal Latitude { get; set; }
        public decimal Longitude { get; set; }
        public DbGeography GeoPoint
        {
                get
                {
                        var res = DbGeography.FromText("POINT(" + this.Longitude.ToString() + " " +
this.Latitude.ToString() + ")", 4326);
                        return res;
                }
                protected set { }
        }
}
```

---
```
<div class="col-md-12 address-request collappse in">
        @Html.EditorFor(model => model.Address, "Address", new { @class = "form-control",  @id = "input-
address", @placeholder = "eg. 1 Martin Pl, Sydney NSW 2000" })
```

```
        @Html.ValidationMessageFor(model => model.Address, "", new { @class = "text-danger" })
</div>


@section Scripts {
 <script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBLAHAeG9pD0ARapCn2AbSAcbYXaX0dUhQ&librar
ies=places"></script>
}

<script>
(function ($) {
        //$('div.datetimepicker').datetimepicker();

        $("input.geocomplete-address").geocomplete({
                componentRestrictions: { country: "AU" },
                details: ".geocomplete-details",
                detailsScope: ".geocomplete-container",
                detailsAttribute: "data-geo"
        });

        $("input.geocomplete-suburb").geocomplete({
                componentRestrictions: { country: "AU" },
                types: ['(regions)'],
                details: ".geocomplete-details",
                detailsScope: ".geocomplete-container",
                detailsAttribute: "data-geo"
        });
})(jQuery);
</script>
```

>Pass data from controller and set value to span 08/16/17

```
JavaScript:
getPatientInfo("@Url.Action("GetPatientInfo")", $("#patientid0").val());

function getPatientInfo(url, id) {
        $("#patientinfo").empty();
        $.ajax({
                type: 'GET',
                url: url,
                dataType: 'json',
                data: {
                        id: id
                },
                success: function (elements) {
                        $.each(elements, function (i, element) {
                                if (element.Text == "pre-address") {
                                        if ($('#address-selecting').prop('checked')) {
                                                document.getElementById(element.Text).innerText =
element.Value;
                                                document.getElementById('full-address').value =
element.Value;
                                        }
                                }
```

```
                                        else
                                                document.getElementById(element.Text).innerText = element.Value;
                                });
                        },
                        error: function (ex) {
                                alert('Failed to retrieve data' + ex);
                        }
                });
                return false;
};

View:
<div class="job-lable">
        <b>Name: </b><span id="pre-name"></span><br />
        <b>Gender: </b><span id="pre-patient-gender"></span><br />
        <b>Age: </b><span id="pre-age"></span><br />
        <b>Interests: </b><span id="pre-interests"></span>
</div>

Controller:
[HttpGet]
public JsonResult GetPatientInfo(string id)
{
        var client = _clientServices.RetrieveClient(this.GetLoggedInUser().ID);
        var patient = this._clientServices.RetrievePatient(client.ID, int.Parse(id));
        DateTime now = DateTime.Today;
        DateTime dob = patient.DoB ?? DateTime.Now;
        int age = now.Year - dob.Year;
        if (now < dob.AddYears(age))
                age--;
        var interests = this._presentationService.GetSubCategoryListByUser("Interest", patient.ID);
        string tag = "";
        foreach (var element in interests)
        {
                tag = element.Name + ", " + tag;
        }

        List<SelectListItem> results = new List<SelectListItem>();
        results.Add(new SelectListItem { Text = "pre-name", Value = patient.FirstName + " " +
patient.LastName });
        results.Add(new SelectListItem { Text = "pre-patient-gender", Value =
this._presentationService.GetGenderByID(patient.GenderID) });
        results.Add(new SelectListItem { Text = "pre-age", Value = age.ToString() });
        results.Add(new SelectListItem { Text = "pre-address", Value = patient.Address.FullAddress });
        results.Add(new SelectListItem { Text = "pre-interests", Value = tag });
        return Json(results, JsonRequestBehavior.AllowGet);
}
```

>Muiltiple dropdownlist 08/16/17

```
<div class="row job-padding" id="selectpatient">
        <div class="col-md-6" id="rootpatient">
                @Html.LabelFor(model => model.PatientId, new { @class = "job-label" })<br />
```

```
                @Html.DropDownListFor(model => model.PatientsID, Model.PatientDropDownList, new
{ @class = "form-control", @id = "patientid0" })
                @Html.ValidationMessageFor(model => model.PatientId, "", new { @class = "text-danger" })
        </div>

        <div class="col-md-6">
                <br />
                <span class="fa fa-plus-circle job-lable more-patient" style="margin-top: 15px; cursor:pointer">
Add another Patient</span>
        </div>
</div>

<div class="row job-padding">
        <div class="col-md-6" id="container"></div>
        <div class="col-md-6" id="remove-patient"></div>
</div>

$(function () {
 $(".more-patient").bind("click", function () {
        var index = $("#container select").length + 1;
        var morepatient = $("#patientid0").clone();
        var itemcount = $("#patientid0 option").length;
        morepatient.attr("id", "patientid" + index);
        morepatient.css("margin-bottom", "4px");

        if (index < itemcount) {
         for (i = 0; i < index; i++) {
                var selectedValue = $("#patientid" + i + " option:selected").val();
                morepatient.find("option[value = '" + selectedValue + "']").remove();
        }
        $("#container").append(morepatient);
        $("#container").fadeIn('fast');
        $('#remove-patient').append('<span class="fa fa-minus-circle job-lable"  style="margin-top: 12px; margin-
bottom:12px; cursor:pointer" OnClick="removePatientBtn_click(this.id)" id="removePatientBtn' + index + '">
Remove additional Patient ' + index + '</span><br>');
                }
        });
});

function removePatientBtn_click(id) {
        var a = id;
        var number = id.substr(id.length - 1);
        $('#remove-patient').find('br').first().remove();
        $("#" + id).remove();

        $("#patientid" + number).remove();
        resetPatientsOrder();
}

function resetPatientsOrder() {
        var additionalPatients = $("#container select").length;
        var index = $("#container select").length;
        var count = 1;
        $("#container select").each(function () {
                $(this).attr("id", "patientid" + count);
```

```
                count++;
        });

        count = 1;
        $('#remove-patient span').each(function () {
                $(this).attr("id", "removePatientBtn" + count);
                $(this).text(" Remove additional Patient " + count);
                count++;
        });
}
```

>Passs model to controller use ajax  08/16/17

http://transtatic.com/blog/2013/04/asp-net-mvc-passing-model-from-ajax-call-into-a-controller-the-easy-way/

View:

```
<fieldset id="infoForm">
   <div style="margin: 1em .5em">
      @Html.LabelFor(model => model.FirstName)
      @Html.TextBoxFor(model => model.FirstName)
   </div>

   <div style="margin: 1em .5em">
      @Html.LabelFor(model => model.LastName)
      @Html.TextBoxFor(model => model.LastName)
   </div>

   <div style="margin: 1em .5em">
      @Html.LabelFor(model => model.Age)
      @Html.TextBoxFor(model => model.Age)
   </div>

   <p>
      <input id="submitButton" type="button" value="Submit Info" />
   </p>
</fieldset>
```

Controller:

```
List<Info> _contactList = new List<Info>();

[HttpPost]
public JsonResult SubmitInfo(Info contactInfo) {
   _contactList.Add(contactInfo);    // Do something with contactInfo

   var response = new Response(true, "Contact Successfully Submitted");
   return Json(response);
}
```

JavaScript:

```
$(function () {
```

```
    var submitButton = $("#submitButton");
    var infoForm = $("#infoForm");

    //  Attach event handler to submit button
    submitButton.click(function () {
        SubmitInfo(infoForm);
    });
});

//  Submits the Info form to Controller
//  formContainer: (jQuery) The form model to submit
function SubmitInfo(formContainer) {
    $.ajax({
        url: "Home/SubmitInfo",
        type: 'post',
        data: formContainer.serialize(),
        success: function (data) {
            if (data.IsSuccess) {
                // Clear the input tags
                formContainer.find("input[type='text']").each(function (i, element) {
                    $(this).val('');
                });
            }

            alert(data.Message);
        },
        error: function (jqXHR, textStatus, errorThrown) {
            alert(errorThrown);
        }
    });
}
```

>Moment.js string to datetimestamp

```
 var dateString = startDate + " " + startTime.format('HH:mm A');
 startDateTime = moment(dateString, "YYYY-MM-DD HH:mm");
```

>JS Dialog:

```
                $("#dialog").dialog({
                        autoOpen: false,
                        height: 350,
                        width: 700,
                        modal: true,
                        buttons: {
                                'Create event': function () {
                                        $(this).dialog('close');
                                },
                                Cancel: function () {
                                        $(this).dialog('close');
                                }
                        },

                        close: function () {
                        }
```

```
                });
```

>FullCalendar get event

https://fullcalendar.io/docs/event_data/events_function/

```
<script>
$(document).ready(function () {

        $('#calendar').fullCalendar({
         events: function (start, end, timezone, callback) {
                $.ajax({
                        url: '@Url.Action("GetCalendarEvents")',
                        dataType: 'json',
                        data: {
                                // our hypothetical feed requires UNIX timestamps
                                start: start.unix(),
                                end: end.unix()
                        },
                        success: function (doc) {
                                var events = [];
                                $(doc).each(function () {
                                        events.push({
                                                title: $(this).attr('title'),
                                                start: $(this).attr('start') // will be parsed
                                        });
                                });
                                callback(events);
                        },
                        error: function () {
                                alert('there was an error while fetching events!');
                        },
                        color: 'yellow',
                        textColor: 'black'

                });
                }
        });

});
</script>

View:
<div>
 <button id="calendarbtn" class="btn btn-default btn-sm" data-toggle="modal" data-target="#calendarModal">
    Calendar</button>
</div>
<!-- Calendar Modal -->
<div class="modal fade" id="calendarModal" tabindex="-1" role="dialog" aria-labelledby="modalLabel">
        <div class="modal-dialog">
                <div class="modal-content">
                  <div class="modal-header">
```

```
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                        <h4 class="modal-title" id="modalLabel">Calendar</h4>
                    </div>
                    <div id="calendar" class="modal-body">
                    </div>
                    <div class="modal-footer">
                      <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
                    </div>
                  </div>
        </div>
</div>

Controler:
[HttpGet]
public JsonResult GetCalendarEvents(double start, double end)
{
        var eventDetails = this._calendarServices.GetCalendarEvents();

        var eventList = from item in eventDetails
                                        select new
                                        {
                                                id = item.ID,
                                                title = item.Title,
                                                start = item.Start.ToString("s"),
                                                end = item.End.ToString("s"),
                                                allDay = false,
                                                editable = false
                                        };

        return Json(eventList.ToArray(), JsonRequestBehavior.AllowGet);
}
```

>Get list from view

```
Model:
public int[] PatientsID { get; set; }
or
public List<int> PatientsID { get; set; }

View:
<div class="col-md-6" id="rootpatient">
        @Html.LabelFor(model => model.PatientId, new { @class = "job-label" })<br />
        @Html.DropDownListFor(model => model.PatientsID, Model.PatientDropDownList,
             new { @class = "form-control", @id = "patientid0" })
        @Html.ValidationMessageFor(model => model.PatientId, "", new { @class = "text-danger" })
</div>

Controler:
var patientsIDs=model.PatientsID;
```

>MVC use Session

```
// set object ot session
Session["info"] = model;
```

```
// get object from session
var modelss = Session["info"];
```

>MVC get Json

```
getPatientInfo("@Url.Action("GetPatientInfo")", $("#patientid0").val());

function getPatientInfo(url, id) {
	$("#patientinfo").empty();
	$.ajax({
		type: 'GET',
		url: url,
		dataType: 'json',
		data: {
			id: id
		},
		success: function (elements) {
			$.each(elements, function (i, element) {
				if (element.Text == "pre-address") {
					if ($('#address-selecting').prop('checked')) {
						document.getElementById(element.Text).innerText =
element.Value;
						document.getElementById('full-address').value =
element.Value;
					}
				}
				else
					document.getElementById(element.Text).innerText = element.Value;
			});
		},
		error: function (ex) {
			alert('Failed to retrieve data' + ex);
		}
	});
	return false;
};

public JsonResult GetPatientInfo(string id)
{
	var client = _clientServices.RetrieveClient(this.GetLoggedInUser().ID);
	var patient = this._clientServices.RetrievePatient(client.ID, int.Parse(id));
	DateTime now = DateTime.Today;
	DateTime dob = patient.DoB ?? DateTime.Now;
	int age = now.Year - dob.Year;
	if (now < dob.AddYears(age))
		age--;
	var interests = this._presentationService.GetSubCategoryListByUser("Interest", patient.ID);
	string tag = "";
	foreach (var element in interests)
	{
		tag = element.Name + ", " + tag;
	}
```

```
        List<SelectListItem> results = new List<SelectListItem>();
        results.Add(new SelectListItem { Text = "pre-name", Value = patient.FirstName + " " +
patient.LastName });
        results.Add(new SelectListItem { Text = "pre-patient-gender", Value =
this._presentationService.GetGenderByID(patient.GenderID) });
        results.Add(new SelectListItem { Text = "pre-age", Value = age.ToString() });
        results.Add(new SelectListItem { Text = "pre-address", Value = patient.Address.FullAddress });
        results.Add(new SelectListItem { Text = "pre-interests", Value = tag });
        return Json(results, JsonRequestBehavior.AllowGet);
// following only work at POST
//      return Json(new SelectList(states, "Value", "Text", JsonRequestBehavior.AllowGet));
}
```

>JS add element in MVC

```
///JS

$('#addPatientBtn').click(function () {
        $.ajax({
                type: 'GET',
                url: '@Url.Content("addpatient")',
                data: "json",
                success: function (data) {
                        $('#divPartial').append(data);
                        $('#divPartial').fadeIn('fast');
                },
                error: function (ex) {
                        alert('Failed to retrieve data' + ex);
                }
        });
});

///view
<div id="divPartial"></div>
<div class="col-md-6">
        <br />
        <span class="fa fa-plus-circle job-lable" style="margin-top: 15px; cursor:pointer" id="addPatientBtn">
Add another Patient</span>
</div>

//return your partial view here
public ActionResult AddPatient()
{
        return PartialView("_AddPatient");
}

///partial view
///_AddPatient.cshtml
<div class="col-md-12 ">
<div class="col-md-6">
        <br />
        <span class="fa fa-plus-circle job-lable" style="margin-top: 15px; cursor:pointer" id="addPatientBtn">
Add another Patient</span>
</div>
```

```
</div>
```

>JS delay 1 sec before get data

```
 $("#full-address").focusout(function () {
        setInterval(delayTimer, 1000);
        function delayTimer() {
        document.getElementById("pre-address").innerText = $("#full-address").val();
        }
});
```

>View Change get data from controller

```
public JsonResult GetPatientInfo(string id)
{
        var client = _clientServices.RetrieveClient(this.GetLoggedInUser().ID);
        var patient = this._clientServices.RetrievePatient(client.ID, int.Parse(id));
        DateTime now = DateTime.Today;
        DateTime dob = patient.DoB ?? DateTime.Now;
        int age = now.Year - dob.Year;
        if (now < dob.AddYears(age))
                age--;
        var interests = this._presentationService.GetSubCategoryListByUser("Interest", patient.ID);
        string tag = "";
        foreach (var element in interests)
        {
                tag = element.Name + ", " + tag;
        }

        List<SelectListItem> states = new List<SelectListItem>();
        states.Add(new SelectListItem { Text = "Address", Value = patient.Address.FullAddress });
        states.Add(new SelectListItem { Text = "Age", Value = age.ToString() });
        states.Add(new SelectListItem { Text = "Name", Value = patient.FirstName + " " + patient.LastName });
        states.Add(new SelectListItem { Text = "Interests", Value = tag });
        return Json(new SelectList(states, "Value", "Text", JsonRequestBehavior.AllowGet));
}


<script>
$(document).ready(function () {

        $("#patientid").change(function () {
                $("#patientinfo").empty();
                $.ajax({
                        type: 'POST',
                        url: '@Url.Action("GetPatientInfo")',
                        dataType: 'json',
                        data: {
                                id: $("#patientid").val()
                        },
                        success: function (states) {
                                $.each(states, function (i, state) {
                                        var el = document.createElement("input");
                                        el.setAttribute("type", "text");
```

```
                                        el.setAttribute("value", state.Value);
                                        el.setAttribute("id", state.Text);
                                        var textbox = document.getElementById("patientinfo");
                                        textbox.appendChild(document.createElement("br"));
                                        textbox.appendChild(el);
                                });
                        },
                        error: function (ex) {
                                alert('Failed to retrieve data' + ex);
                        }
                });
                return false;
        })
});
</script>
```

>Pass data from Viewbag into a js variable

```
<script type="text/javascript">
   var foo = @Html.Raw(Json.Encode(ViewBag.FooBar))
</script>
```

or

```
@model MyViewModel
<script type="text/javascript">
   var foo = @Html.Raw(Json.Encode(Model))
</script>
```

```
                >model

                <button type="button" class="btn btn-default" id="btnDetails">Details</button>
                        <script>
                                $("#btnDetails").click(function () {
                                        $('#DetailsPopup').appendTo("body").modal('show');
                                });
                        </script>
                <div class="modal fade" id="DetailsPopup" tabindex="-1" role="dialog" data-backdrop="static"
                        data-keyboard="true" aria-hidden="true">
                                <div class="modal-dialog">
                                        <div class="modal-content">
                                                @Html.Partial("_JobPreview", Model);
                                        </div>
                                </div>
                        </div>
```

>Pass model to js

```
                                var theModel =
@Html.Raw(Newtonsoft.Json.JsonConvert.SerializeObject(Model));
```

>JS dialog

```
<script type="text/javascript">
        $(function () {
                $('.modal11').click(function () {
                        $('<div />').appendTo('body').dialog({
                                close: function (event, ui) {
                                        dialog.remove();
                                },
                                modal: true
                        }).load(this.href, {});

                        return false;
                });
        });
</script>
@Html.ActionLink("sowl", "modal", "job", null, new { @class = "modal11 btn btn-default btn-sm" })
```

>Jquery datepicker error

**Solution 1**

```
<script>
        jQuery(function ($) {

          jQuery.validator.methods.date = function (value, element) {
                var isChrome = /Chrome/.test(navigator.userAgent) && /Google Inc/.test(navigator.vendor);
                var isSafari = /Safari/.test(navigator.userAgent) && /Apple Computer/.test(navigator.vendor);
                if (isSafari || isChrome) {

                  var d = value.split("/");
                  return this.optional(element) || !/Invalid|NaN/.test(new
Date(/chrom(e|ium)/.test(navigator.userAgent.toLowerCase()) ? d[1] + "/" + d[0] + "/" + d[2] : value));

        ////var d = new Date();   //have error
        ////return this.optional(element) || !/Invalid|NaN/.test(new Date(d.toLocaleDateString(value)));

                } else {
                        return this.optional(element) || !/Invalid|NaN/.test(new Date(value));
                }
          };
        });
</script>
```

**Solution 2**

```
$(function() {
   $.validator.methods.date = function (value, element) {
      if ($.browser.webkit) {

         //ES - Chrome does not use the locale when new Date objects instantiated:
         var d = new Date();

         return this.optional(element) || !/Invalid|NaN/.test(new Date(d.toLocaleDateString(value)));
      }
      else {

         return this.optional(element) || !/Invalid|NaN/.test(new Date(value));
      }
   };
```

});
**Solution 3**
$("input[data-val-date]").removeAttr("data-val-date");

>Google Apps Help
https://developers.google.com/google-apps/
Https://sites.google.com/a/sisd.cc/google-apps-help/

>Error for google calendar: frame because it set 'X-Frame-Options' to 'sameorigin'.

```
gapi.auth.authorize({client_id: clientId, scope: scopes, immediate: true}, handleAuthResult);
```

Change here to >> immediate: false

>TimeZone
https://www.iana.org/time-zones
>> Data Only Distribution >>zone.tab

| #code | coordinates | TZ | comments |
|---|---|---|---|
| AU | -4253+14719 | Australia/Hobart | Tasmania (most areas) |
| AU | -3956+14352 | Australia/Currie | Tasmania (King Island) |
| AU | -3749+14458 | Australia/Melbourne | Victoria |
| AU | -3352+15113 | Australia/Sydney | New South Wales (most areas) |
| AU | -3157+14127 | Australia/Broken_Hill | New South Wales (Yancowinna) |
| AU | -2728+15302 | Australia/Brisbane | Queensland (most areas) |
| AU | -2016+14900 | Australia/Lindeman | Queensland (Whitsunday Islands) |
| AU | -3455+13835 | Australia/Adelaide | South Australia |
| AU | -1228+13050 | Australia/Darwin | Northern Territory |
| AU | -3157+11551 | Australia/Perth | Western Australia (most areas) |
| AU | -3143+12852 | Australia/Eucla | Western Australia (Eucla) |

>URL customer Setting
RouteConfig.cs

```
routes.AppendTrailingSlash = true;
routes.LowercaseUrls = true;
```

>Location for:

```
DataStore = new FileDataStore("Calendar.Api.Auth.Store")
```

C:\Users\Steven\AppData\Roaming\

>Add google login
VS Install-Package

Google.Apis.Calendar.v3
Microsoft.Owin.Security.Google

Login Google Console to set : https://console.developers.google.com/apis

add google+api and google calendar as enable,

add OAuth 2.0 client Ids, and get:
Client ID: 434717621078-unpnsvsmjhn4tub93463lmkcs96q8cmi.apps.googleusercontent.com
Client secret: dCCnPhZ1eC_khC7R3-948CHL
In  Authorized redirect URIs,  Add: http://localhost:5000/signin-google

Doc:  https://developers.google.com/api-client-library/dotnet/apis/calendar/v3
Google Example:  https://github.com/google/google-api-dotnet-client-samples

>Change port in VS  07/18/17
To specify a port for a Web application project that uses IIS Express
In Solution Explorer, right-click the name of the application and then select Properties. Click the Web tab.
In the Servers section, under Use Local IIS Web server, in the Project URL box change the port number.
To the right of the Project URL box, click Create Virtual Directory, and then click OK.
In the File menu, click Save Selected Items.
To verify the change, press CTRL+F5 to run the project. The new port number appears in the address bar of the browser.

>Add WebApi2 controller in MVC
At Controllers, mouse right key, Add->Controller, select Web API 2 Controller-Empty

Add or Generated: App_Start/WebApiConfig.cs

```
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        config.MapHttpAttributeRoutes();

        config.Routes.MapHttpRoute(
            name: "DefaultApi",
            routeTemplate: "api/{controller}/{id}",
            defaults: new { id = RouteParameter.Optional }
        );
    }
}
```

Install-Package Microsoft.AspNet.WebApi.WebHost  for GlobalConfiguration.
Register: Global.asax.cs

```
protected void Application_Start()
{
    // add this font than others
    GlobalConfiguration.Configure(WebApiConfig.Register);   <<<<
}
```

Autofac:

Add Autofac.WebApi2 refference from NuGet
Add "<<<<" part in App_Start/Startup.Container.cs

```
// Register MVC Controllers
```

```
builder.RegisterControllers(assembly);
builder.RegisterApiControllers(Assembly.GetExecutingAssembly());  <<<<

DependencyResolver.SetResolver(new AutofacDependencyResolver(container));
GlobalConfiguration.Configuration.DependencyResolver =
  new AutofacWebApiDependencyResolver((IContainer)container);    <<<<
```

Using:  https://localhost:44301/api/getfile

```
        [Route("customers/orders")]
        [HttpGet]
        public HttpResponseMessage calender()
        {
                var xmlString = "<xml><name>Some XML</name></xml>";
                var result = Request.CreateResponse(HttpStatusCode.OK);
                result.Content = new StringContent(xmlString, Encoding.UTF8, "application/xml");
                result.Content.Headers.ContentDisposition = new
ContentDispositionHeaderValue("attachment")
                {
                        FileName = "test.xml"
                };

                return result;
        }

        // GET api/calendar/{id}
        public IHttpActionResult Get(string id)
        {
                return Ok("It works! Your id is " + id);
        }

        [Route("api/getfile")]
        [HttpGet]
        public HttpResponseMessage GetFile()
        {
                var result = this._calendarServices.CreateIcsFile();

        //      return Request.CreateResponse(result);
        return result;

        }
```

>Deploy to Azure

==========

1. FileZilla Setting:

==========

Host: waws-prod-sy3-011.ftp.azurewebsites.windows.net/site/wwwroot
Protocol: FTP - File Transfer Protocol
Encryption: Only use plain FTP

Logon Type: Normal

User: abilityfirst-test\$abilityfirst-test
Password: 9nnuaQ1BZ3wpFz4AyCstEerpTjmppZdhoR7edC4w0H5T6HXlwkHKWwgttyGi

==========

Please upload deploy files into the following folder:

/site/wwwroot

==========

==========
2. Connection String
==========
```
<connectionStrings>
  <add name="MyAbilityFirstDbContext"
       connectionString="Data Source=tcp:mafdb.database.windows.net,1433;
    Initial Catalog=MAFTest;
       User Id=myabilityfirst@mafdb.database.windows.net;
    Password=HappyCh1ck3n;
    MultipleActiveResultSets=true;"
       providerName="System.Data.SqlClient"
  />
</connectionStrings>
```
==========

==========
3. URL
==========
http://abilityfirst-test.azurewebsites.net

>Change Azure SQL Database name
>>Can not change the name include “-”
The following example changes the name of the AdventureWorks2012 database to Northwind.

```
USE master;
GO
ALTER DATABASE AdventureWorks2012
Modify Name = Northwind ;
GO
```

>SMS verification

https://docs.microsoft.com/en-us/aspnet/mvc/overview/security/aspnet-mvc-5-app-with-sms-and-email-two-factor-authentication

The 2FA codes are generated using Time-based One-time Password Algorithm and codes are valid for six minutes.

AccountController.cs

```
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    if (!ModelState.IsValid)
    {
        return View(model);
    }

   // Require the user to have a confirmed email before they can log on.
    var user = await UserManager.FindByNameAsync(model.Email);
    if (user != null)
    {
      if (!await UserManager.IsEmailConfirmedAsync(user.Id))
      {
        ViewBag.errorMessage = "You must have a confirmed email to log on.";
        return View("Error");
      }
    }
    // This doen't count login failures towards lockout only two factor authentication
    // To enable password failures to trigger lockout, change to shouldLockout: true
    var result = await SignInManager.PasswordSignInAsync(model.Email, model.Password,
      model.RememberMe, shouldLockout: false);
    switch (result)
    {
        case SignInStatus.Success:
            return RedirectToLocal(returnUrl);
        case SignInStatus.LockedOut:
            return View("Lockout");
        case SignInStatus.RequiresVerification:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUrl });
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "Invalid login attempt.");
            return View(model);
    }
}
```

>Admin user:

/account/co-ordinator-register/
select "Domestic Staff" register
go to AspNetRoles Change "Domestic Staff" to "Admin"
/manage/index/
Enabled  for Two-Factor Authentic
=>> when login, it need email or phone to receive a security code, such as: 073316

---

>Coordinator user:

1. /account/co-ordinator-register/

2. select "Co-ordinator" register

>FileZilla Azure FTP Setting

Web App:
Host: waws-prod-sy3-011.ftp.azurewebsites.windows.net
Protpcol: FTP – File Transfer Protocol
Encryption: Only use plain FTP (insecure)
Logon Type: Normal
 ==> *Get publish profile to get username and password*
User: abilityfirst\$abilityfirst
Password:

Mobile App:
Host: ftp://waws-prod-sy3-011.ftp.azurewebsites.windows.net/site/wwwroot
User: myablityfirstmobile\$myablityfirstmobile

>Google Cloud Console
https://console.developers.google.com/cloud-resource-manager?pli=1

>X-Frame-Options DENY in a Chrome extension?

Chrome offers the webRequest API to intercept and modify HTTP requests. You can remove the X-Frame-Options header to allow inlining pages within an iframe.

```
chrome.webRequest.onHeadersReceived.addListener(
   function(info) {
      var headers = info.responseHeaders;
      for (var i=headers.length-1; i>=0; --i) {
         var header = headers[i].name.toLowerCase();
         if (header == 'x-frame-options' || header == 'frame-options') {
            headers.splice(i, 1); // Remove header
         }
      }
      return {responseHeaders: headers};
   },
   {
      urls: [ '*://*/*' ], // Pattern to match all http(s) pages
      types: [ 'sub_frame' ]
   },
   ['blocking', 'responseHeaders']
);
```

In the manifest, you need to specify the webRequest and webRequestBlocking permissions, plus the URLs patterns you're intending to intercept.

>Test Post request use Postman in chrome
www.getpostman.com
https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop?hl=en

>keep free Azure app alive
Windows Azure Web Site in free or shared mode, after some time of inactivity (between 5 and 20 minutes) your web application will shut down to free up resources on the server and this one of the reasons why your site could

seem slow (the first user visiting the site after the shut down will need to wait for the application to start again). This is something you'll typically see with a low traffic site.

http://fabriccontroller.net/job-scheduling-in-windows-azure/

>resolve issues when open a existing solution
Solution Explorer->right mouse key->clean solution, rebuild solution
go to project References->right mouse key->Manage NuGet Package →
1. [restore]
2. [update](some time need select update, if no error, don't need update)
3. the folder name too long, need to change folder name

>Add storage accounts by using Server Explorer in VS
To attach an existing storage account by using Server Explorer
In Server Explorer, open the shortcut menu for the Azure storage node, and then choose Attach External Storage.

https://abilityfirststorage.queue.core.windows.net/

AccountName=abilityfirststorage
AccountKey=r5pUS309VutQBo5FDuP1pmY5X+BNe7jzuEEaK3w7giGaqPdFN3JxMwLjfs6+UjqR4oGJlT/HAVVUv2g3a5xj0A==

> Azure blob
Miscrosoft Azure Storage Explorer
myabilityfirst@gmail.com
HappyCh1ck3n

DefaultEndpointsProtocol=https;AccountName=abilityfirststorage;AccountKey=r5pUS309VutQBo5FDuP1pmY5X+BNe7jzuEEaK3w7giGaqPdFN3JxMwLjfs6+UjqR4oGJlT/HAVVUv2g3a5xj0A==;EndpointSuffix=core.windows.net

>smb using

IOS:
smb://172.17.200.5/Storage
user: stephen@cenozoic.com.au
password: Ceno2937

win:
\\172.17.200.5\
user: stephen
password: Ceno2937

>Error for Maping

need register in dbcontext, for example:

```
modelBuilder.Entity<UserAttachment>()
```

>chrome display video

```
var isChrome = !!window.chrome && !!window.chrome.webstore;
  if (isChrome) {
        $("#videoId").replaceWith($('<video id="videoId" width="100%" autoplay loop><source src=' + recipient
+ ' type="video/webm"></video>'));
        }

<video id="videoId" width="550" controls="controls">
        <source src="" type="video/mp4; codecs='avc1.42E01E, mp4a.40.2'" class="modal-videoplayer">
        <source src="" type="video/webm; codecs='vp8, vorbis'" class="modal-videoplayer">
        <source src="" type="video/ogg; codecs='theora, vorbis'" class="modal-videoplayer">
</video>
```

>Git

```
  git checkout -b feature/new-branch          (create a new branch)
  git push (-u) origin feature/new-branch     (push to remote server)

 git status                                   (current branch action status)
 git branch                                   (list local branch and current branch)
 git branch -a                                (list all branch)
 git checkout  master                         (switch to master)
```

>Add new branch in VS

```
 In visual studio
 Create a new branch:
        > at bottom-right select master, than select new branch
        a. input new branch name such as: feature\new-branch
        b. select origin/master
c. don't select “Track remote branch”
d. press Create Branch
```

>Error fix  for – Microsoft.SqlServer.Types' version 10 or higher could not be found.

http://stackoverflow.com/questions/13174197/microsoft-sqlserver-types-version-10-or-higher-could-not-be-found-on-azure

```
 Install-Package Microsoft.SqlServer.Types
Install Microsoft System CLR Types for SQL Server 2012 (do not know why?)
(X64 :http://go.microsoft.com/fwlink/?LinkID=239644&clcid=0x409 )
Do not need add any as below in global.asax.cs
now migration work
```

sqlserver.type error, add following code in global.asax.c

```
        SqlServerTypes.Utilities.LoadNativeAssemblies(Server.MapPath("~/bin"));

SqlProviderServices.SqlServerTypesAssemblyName =
   "Microsoft.SqlServer.Types, Version=14.0.0.0, Culture=neutral, PublicKeyToken=89845dcd8080cc91";
```

>Link in view

```
<td>
@{
        if (item.JobID != 0)
        {
                var jobTitle = "Communication base on job '" + item.JobTitle + "'";
```

```
                    <span title="@jobTitle" data-toggle="tooltip" data-placement="top"> <a
href="../../job/details/@item.JobID">#@item.JobID</a></span>
                    @Html.Partial("_JobStatus", @item.JobStatus)
          }
          else
          {
          <span class="fa fa-weixin" aria-hidden="true" style="color:skyblue" title="Normal communication">
</span>
          }
}
</td>

>Reload page
<script>
          function reloadData() {
                    location.reload();
          }
</script>
 <button onclick="reloadData()" class="btn btn-info btn-sm" title="reload this page to get new message">Reload
page</button>

>merge two list in one

var newMessagesFromCarer =
newMessagesFromCarerWithoutJobID.Concat(newMessagesFromCarerBaseOnJobID).ToList();

>Entity Framework Code First Data Annotations
https://msdn.microsoft.com/en-us/data/jj591583.aspx#Index

[Required]
[MaxLength(10),MinLength(5)]
[MaxLength(10, ErrorMessage="BloggerName must be 10 characters or less"),MinLength(5)]



>Partial View

  @Html.Action("_Details", "Job", new { id = Model.JobID })    //have controller
  { return PartialView(vmList); }             //in controller

  @Html.Partial("_JobStatus", @Model.JobStatus)          //with out controller


>Chrome not respond
cmd
ipconfig/flushdns
netsh winsock reset

download
ccleaner cnet


>Customer validation
```

```
<div class="form-group">
    @Html.LabelFor(model => model.OverallScore, new { @class = "col-md-2 control-label" })
    <div class="col-md-10">
                <span class="field-validation-valid" style="font-size:x-large; color:orange; cursor:pointer">
                    @Html.BootstrapRating(model => model.OverallScore)
                </span>
                @Html.ValidationMessageFor(model => model.OverallScore, "", new { @class = "text-danger",
                  @id = "OverallScoreValidation", @required = true })
    </div>
</div>

<div class="form-group">
    <div class="col-sm-offset-5 col-md-7" style="padding-top:7px;">
                <input type="submit" value="Submit" class="btn btn-primary" />
                @Html.ActionLink("Cancel", "Rating", "Client", null, new { @class = "btn btn-default" })
    </div>
</div>

<script>
    $("form").submit(function (e) {
                var rated = true;
                e.preventDefault();
                $('.container').find('.rating').each(function () {
                    var ratevalue = $(this).val();
                    var name = $(this).attr('name');
                    if (name != null && ratevalue == "") {
                        $('#' + name + "Validation").html("Please rate this !");
                        rated = false;
                    }
                    else {
                        $('#' + name + "Validation").html("");
                    }
                });
                if (rated)
                    this.submit();
    });
</script>

--------------------------------

                [HttpGet, Route("patient/new")]
                public ActionResult NewProfile()
                {
                    var client = this.GetLoggedInUser() as Client;
                    PatientDetailsViewModel vm = mapPatientToVM(new Patient(client.ID));
                    ViewBag.CheckLocation = Url.Action("CheckLocation", "Client");
                    return View(vm);
                }



<div class="form-group">
@Html.LabelFor(model => model.Suburb, htmlAttributes: new { @class = "control-label col-md-3" })
<div class="col-md-9">
```

```
  @Html.TextBoxFor(model => model.ResidentialSuburbID, new { @class = "form-control", @id = "resientialId",
@style = "display: none" })
     <div class="geocomplete-container">
          @Html.TextBox("resiential", null, new { @class = "form-control geocomplete-suburb" })
          @Html.ValidationMessageFor(model => model.ResidentialSuburbID, "", new { @class = "text-danger",
@id = "resientialValidation" ,@required=true})
          <div class="geocomplete-details resiential">
            <input type="text" name="profileSuburb" data-geo="locality" value="" />
            <input type="text" name="profileState" data-geo="administrative_area_level_1" value="" />
            <input type="text" name="profilePostcode" data-geo="postal_code" value="" />
            <input type="text" name="profileLat" data-geo="lat" value="" />
            <input type="text" name="profileLng" data-geo="lng" value="" />
          </div>
  </div>
</div>
</div>

jQuery.noConflict()(function ($) {

    $(".geocomplete-suburb").geocomplete()
          .bind("geocode:result", function (event, result) {
          {
                              var locationArray = new Array();
                              var dataError = false;
                              var id = $(this).attr('id');
                              $('.' + id).find('input').each(function () {
                                        if (locationArray.length < 5) {
                                                  var inputName = $(this).attr('name');
                                                  var addressValue = $('.' + id + " input[name=" + inputName +
"]").val();

                                                  if (addressValue === "")
                                                            dataError = true;
                                                  locationArray.push(addressValue);
                                        }
                              });

                              var locationId = $(this).attr('name') + "Id";
                              var validationName = $(this).attr('name') + "Validation";
                              if (!dataError) {
                                        getIdFromServer(locationArray, locationId, validationName);
                              }

                              $('.geocomplete-details').find('input').each(function () {
                                        $("input[name=" + $(this).attr('name') + "]").val("");
                              });
          }

          });

       function getIdFromServer(locationArray, locationId, validationName) {
                    $.ajax({
                              dataType: 'json',
                              url: $('#checkLocationId').prop("value"),
                              type: 'PUT',
                              data: { 'location': locationArray },
```

```
                        success: function (result) {
                                $('#' + locationId).val(parseInt(result.message));
                                $('#' + validationName).html("");
                        },
                        error: function (result) {
                                $('#' + validationName).html("The address is not found");
                        }
                });
        }
});
```

>bootstrap-rating.mvc

VS, NuGet find bootstrap-rating, install.

```
@Html.BootstrapRating()
    .Class("rating-tooltip")
```

>Resize image and Upload file using jcrop  sample AvatarUpload
https://github.com/MiroJ

> bit of code
https://bitsofco.de/the-accessibility-cheatsheet/
http://www.w3.org/TR/WCAG20/#context-sensitivehelpdef

>SQL Geo distance

linq:

```
Distance =DbGeography.FromText("POINT(" +c.Address.Longitude.ToString()  + " " +
                                        c.Address.Latitude.ToString()
+ ")").Distance(DbGeography.FromText("POINT(" +   cw.Address.Longitude.ToString() + " "
+cw.Address.Latitude.ToString()
                                        + ")")).ToString(),
```

Microsoft.SqlServer.Types

```
                        var carerGeoPoint = getGeoPoint(careWorker.Address);
                        if (!clientGeoPoint.Lat.IsNull && !carerGeoPoint.Lat.IsNull)
                        {
                                var distanceDouble = carerGeoPoint.STDistance(clientGeoPoint) / 1000;
                                distance = distanceDouble.Value.ToString("F1", CultureInfo.InvariantCulture) +
" km";
                        }

                private SqlGeography getGeoPoint(Address address)
                {
                        var lat = Convert.ToDouble(address.Latitude);
                        var lang = Convert.ToDouble(address.Longitude);
                        var point = SqlGeography.Point(lat, lang, 4326);
                        return point;
                }
```

deploy Azure

```
<dependentAssembly>
   <assemblyIdentity name="Microsoft.SqlServer.Types"
                     publicKeyToken="89845dcd8080cc91"
                     culture="neutral" />
   <bindingRedirect oldVersion="10.0.0.0" newVersion="11.0.0.0" />
</dependentAssembly>
```

>get current url

```
String strPathAndQuery = HttpContext.Current.Request.Url.PathAndQuery;
String strUrl = HttpContext.Current.Request.Url.AbsoluteUri.Replace(strPathAndQuery, "/");
```

result in these:
"http://localhost:1234/"
"https://www.something.com/"

>RazorEngine
http://mehdi.me/generating-html-emails-with-razorengine-introduction/

>Razor

http://www.knowsh.com/Notes/NotesSearch/NotesDetail/70085/Razor,-Webform-engine,-HTML-Helpers,-Url-Helper,-Validation-Summary,-AJAX-Helper,-unobstrusive-AJAX,-cross-domain-AJAX,-layout-in-MVC

>Resolving the conflict

entry repository directory

Check out and pull the most recent version of the destination (master) branch from the remote.
Git checkout master
git pull origin master

Check out the source (new-names) branch.
git checkout feature/booking-interface

Merge the  master  branch into the new-names branch.

git merge master

*Sometime need comment, after write comment, Press Esc on keyboard → :W  (save)→ :Q (quit)*

see different use merge and push

fix migration,  in VS, press “merge”, select change, press “target”

>Google place API sample

https://google-developers.appspot.com/maps/documentation/javascript/examples/full/places-autocomplete-addressform

>Send Array to Constructor use Ajex

```
var locationArray = new Array();
var dataError = false;
$('.geocomplete-details').find('input').each(function () {
	if (locationArray.length < 5) {
		var addressValue = $("input[name=" + $(this).attr('name') + "]").val();
		if (addressValue === "")
			dataError = true;

		locationArray.push(addressValue);
 //	locationArray = { address, suburb, latitude, longitude,newrange}; <<<< have error

	}
});

var locationId = $(this).attr('name') + "Id";
if (!dataError) {
	getIdFromServer(locationArray, locationId);
}


	function getIdFromServer(locationArray, locationId) {
		$.ajax({
			dataType: 'application/json',
			url: $('#checkLocationId').prop("value"),
			type: 'GET',
			data: { 'location': locationArray },
			traditional: true,			//<<< need this
			success: function (result) {
				$('#' + locationId).val(reslult.id);
	//locationArray = [];		<<clear array
			},
			error: function () {
				alert("error");
			}
		});
	}

		[HttpGet]
		public virtual ActionResult CheckLocation(string[] location)
		{
			//profileSuburb,profileState,profilePostcode,profileLat,profileLng
			var locationId = this._categoryServices.GetLocationId(location);
			if (locationId == 0)
				return Json(new { message = "Error", id = 0 }, "text/html");
			else
				return Json(new { message = "Found", id = locationId }, "text/html");
		}

>>>Javascrpt
jQuery.noConflict()(function ($) {

	$(".geocomplete-suburb").geocomplete()
		.bind("geocode:result", function (event, result) {
			{
```

```
                                var locationArray = new Array();
                                var dataError = false;
                                var id = $(this).attr('id');
                                $('.' + id).find('input').each(function () {
                                        if (locationArray.length < 5) {
                                                var inputName = $(this).attr('name');
                                                var addressValue = $('.' + id + " input[name=" + inputName +
"]").val();

                                                if (addressValue === "")
                                                        dataError = true;
                                                locationArray.push(addressValue);
                                        }
                                });

                                var locationId = $(this).attr('name') + "Id";
                                var validationName = $(this).attr('name') + "Validation";
                                if (!dataError) {
                                        getIdFromServer(locationArray, locationId, validationName);
                                }

                                $('.geocomplete-details').find('input').each(function () {
                                        $("input[name=" + $(this).attr('name') + "]").val("");
                                });
                        }

                });

        function getIdFromServer(locationArray, locationId, validationName) {
                $.ajax({
                        dataType: 'json',
                        url: $('#checkLocationId').prop("value"),
                        type: 'PUT',
                        data: { 'location': locationArray },
                        success: function (result) {
                                $('#' + locationId).val(parseInt(result.message));
                                $('#' + validationName).html("");
                        },
                        error: function (result) {
                                $('#' + validationName).html("The address is not found");
                        }
                });
        }
});
```

>Get Latitude and Longitude using Geocode by Awesome Table
Go to Google Sheets, Add-ons, add “Geocode by Awesome Table ”
Add-ons-> Geocode by Awesome Table->Starting Geocode, select address columns
Geocode code, get Latitude and Longitude

https://chrome.google.com/webstore/detail/geocode-by-awesome-table/cnhboknahecjdnlkjnlodacdjelippfg?hl=en

>Add library
Bootstrap.v3.Datetimepicker

bootstrap-toggle.less  =>toggle ui
geocomplete  - jQuery Geocoding and Places Autocomplete Plugin
http://ubilabs.github.io/geocomplete/
Moment.js  =>date format

>Update file to Azure Storage Blobs

In NuGet to install:
WindowsAzure.Storage
Microsoft.WindowsAzure.ConfigurationManager

```
<link href="~/Content/themes/base/jquery-ui.css" rel="stylesheet" />
<script src="~/Scripts/jquery-ui-1.12.1.min.js"></script>
<script src="~/Scripts/jQuery.FileUpload/jquery.fileupload.js"></script>
<script src="~/Scripts/jQuery.FileUpload/jquery.fileupload-ui.js"></script>
<script src="~/Scripts/jQuery.FileUpload/jquery.fileupload-process.js"></script>
<script src="~/Scripts/Shared/uploadpicture.js"></script>

<!-- begin upload picture -->
<div class="form-group">
   @Html.LabelFor(model => model.PictureURL, new { @class = "control-label col-md-3" })
    <div class="col-md-6">
            <input type="file" name="files" id="uploadPicture" accept="image/*" capture class="btn btn-default btn-
sm" style="width:300px" />
            <input id="uploadURL" value="@ViewBag.PathUpload" hidden="hidden" />
            <input id="deleteURL" value="@ViewBag.PathDelete" hidden="hidden" />
            <div class="progress" style="width:300px">
            <div id="uploadPictureBar" class="progress-bar" role="progressbar" aria-valuenow="0" aria-
valuemin="0" aria-valuemax="100" style="width: 0%;">
             <div class="sr-only">0% complete</div>
             </div>
             <span style="color:red; font-size:small">
                        * Allowed upload file JPG/GIF/BMP/PNG, Max.2MB
             </span>
            </div>
    <input type="button" name="cancelUploadPicture" id="cancelUploadPicture" class="btn btn-default btn-sm"
value="Remove" disabled />
            </div>
            @if (!String.IsNullOrEmpty(Model.PictureURL))
            {
               ViewBag.URL = @Url.Content(Model.PictureURL);
            }
            else
            {
               ViewBag.URL = "";
            }
            <div class="col-md-2 ">
               <img src="@ViewBag.URL" id="pictureFileId" height="180" width="180" border="0">
            </div>

            <div class="col-md-10" hidden="hidden">
               @Html.EditorFor(model => model.PictureURL, new { htmlAttributes = new { @class = "form-control",
@id = "pictureRecord", @readonly = "readonly", @value = @ViewBag.pictureURL } })
```

```
            @Html.ValidationMessageFor(model => model.PictureURL, "", new { @class = "text-danger" })
        </div>
</div>
<!-- end of upload picture -->


//upload file to Azure storage blob
[HttpPost]
public virtual ActionResult UploadFileToAzure()
{
    string path = ConfigurationManager.AppSettings["uploadAzurePath_Job"];
    HttpPostedFileBase file = Request.Files[0];

    string url = this._uploadServices.UploadToAzureStorage(file, path);

    return attachmentProcess(url);
}

[HttpDelete]
public virtual ActionResult DeleteFileFromAzure(string fileName)
{
    string path = ConfigurationManager.AppSettings["uploadAzurePath_Job"];
    if (this._uploadServices.DeleteFromAzureStorage(fileName, path))
      return Json(new { message = "The file has delete !" }, "text/html");
   else
     return Json(new { message = "Error" }, "text/html");
}

private JsonResult attachmentProcess(string url)
{
        bool isUploaded = false;
        if (url != null)
        {
                isUploaded = true;
                string message = "100% complete";

                return Json(new
                {
                        statusCode = 200,
                        status = "File uploaded.",
                        file = url,
                        isUploaded = isUploaded,
                        message = message
                }, "text/html");

        }
        else
        {
                string message = "Error";
                return Json(new
                {
                        statusCode = 500,
                        status = "Error uploading image.",
                        file = string.Empty,
                        isUploaded = isUploaded,
```

```
                        message = message
                }, "text/html");
        }
}

jQuery.noConflict()(function ($) {
        $(document).ready(function () {
                $('#uploadPicture').fileupload({
                        dataType: 'json',
                        url: $('#uploadURL').prop('value'),
                        autoUpload: true,

                        add: function (e, data) {
                                var uploadErrors = [];
                                var fileType = data.files[0].name.split('.').pop(), allowdtypes =
'jpeg,jpg,png,gif,JPEG,JPG,PNG,GIF';
                                if (allowdtypes.indexOf(fileType) < 0) {
                                        uploadErrors.push('Invalid file type, aborted');
                                }
                                //2000000 --> 2M
                                if (data.originalFiles[0]['size'] > 2000000) {
                                        uploadErrors.push('File size is too big');
                                }
                                if (uploadErrors.length > 0) {
                                        alert(uploadErrors.join('\n'));
                                } else {
                                        data.submit();
                                }
                        },
                        start: function (e, data) {
                                $('#uploadPictureBar').css('width','5%');
                                $('#uploadPictureBar').text('5% complete');
                        },
                        progress: function (e, data) {
                                var progress = parseInt(data.loaded / data.total * 100, 10);

                                if (progress < 80) {
                                        $('#uploadPictureBar').css('width', progress + '%');
                                        $('#uploadPictureBar').text(progress + '% complete');
                                }
                        },
                        fail: function (event, data) {
                                if (data.files[0].error) {
                                        var thisId = $(this).attr('id');
                                        var resultId = 'result' + thisId[thisId.length - 1];
                                        $('#' + resultId).html(data.files[0].error);
                                }
                        },
                        done: function (e, data) {
                                var thisId = $(this).attr('id');
                                var resultId = 'result' + thisId[thisId.length - 1];
                                var cancelId = 'cancelUpload' + thisId[thisId.length - 1];
                                var barId = 'bar' + thisId[thisId.length - 1];
                                if (data.result.isUploaded) {
```

```
                                $('#uploadPictureBar').css('width', '100%');
                                $('#uploadPictureBar').text('100% complete');
                                $('#cancelUploadPicture').prop('disabled', false);

                                $('.progress').hide();
                                $('#uploadPicture').hide();
                                $('#uploadPictureBar').hide();
                                $('#cancelUploadPicture').show();

                                //write in EditorFor
                                var fileRecordId = 'fileRecord' + thisId[thisId.length - 1];

                                $('#pictureRecord').attr('value', data.result.file);

                                $('#' + thisId).prop('disabled', true);
                                //show picture
                                //readURL(this);
                                $('#pictureFileId').attr('src', data.result.file);
                        }
                        else {
                                $('#uploadPictureBar').css('width', '0%');
                                $('#uploadPictureBar').text('0% complete');
                        }

                }
        });

        //check upload file status

        var fileName = $('#pictureRecord').val();

        if (fileName === null || fileName === '') {
                $('#cancelUploadPicture').prop('disabled', true);

                $('#uploadPicture').prop('disabled', false);
                $('#uploadPictureBar').css('width', '0%');
                $('#uploadPictureBar').text('0% complete');

                $('.progress').show();
                $('#uploadPicture').show();
                $('#uploadPictureBar').show();
                $('#cancelUploadPicture').hide();
        }
        else {
                $('#cancelUploadPicture').prop('disabled', false);
                $('#uploadPicture').prop('disabled', true);
                $('#uploadPictureBar').css('width', '100%');
                $('#uploadPictureBar').text('100% complete');

                $('.progress').hide();
                $('#uploadPicture').hide();
                $('#uploadPictureBar').hide();
                $('#cancelUploadPicture').show();
        }
});
```

```
	//preview picture
	function readURL(input) {
		if (input.files && input.files[0]) {
			var reader = new FileReader();

			reader.onload = function (e) {
				$('#pictureFileId').attr('src', e.target.result);
			};

			reader.readAsDataURL(input.files[0]);
		}
	}

	//clean file name
	$('#cancelUploadPicture').click(function () {
		var fileName = $('#pictureRecord').val();
		// delete file from server
		$.ajax({
			dataType: 'json',
			url: $('#deleteURL').prop('value'),
			type: 'DELETE',
			data: { 'fileName': fileName },
			success: function (result) {
				///		alert(result.message);
				$('#uploadPictureBar').css('width', '0%');
				$('#uploadPictureBar').text('');
				//clear file name
				$('#pictureRecord').attr('value', '');
				$('#uploadPicture').prop('disabled', false);
				//disable cancel button
				$('#cancelUploadPicture').prop('disabled', true);
				$('#pictureFileId').attr('src', '');

				$('.progress').show();
				$('#uploadPicture').show();
				$('#uploadPictureBar').show();
				$('#cancelUploadPicture').hide();
			},
			error: function () {
				alert('error');
			}
		});

	});

});

//file type
function isImage(filename) {
	var ext = getExtension(filename);
	switch (ext.toLowerCase()) {
		case 'jpg':
		case 'gif':
		case 'bmp':
```

```
                case 'png':
                        //etc
                        return true;
        }
        return false;
}

function getExtension(filename) {
        var parts = filename.split('.');
        return parts[parts.length - 1];
}
```

>ICollection<T>, IList<T>, IEnumerable<T>

ICollection<T>: base interface
IEnumerable<T>: can not change value,allows the largest possible variety of sources
IList<T> : have index, can not Add(item), Remove(item)

>
Abstract methods have no implementation, so the method definition is followed by a semicolon (:) instead of a normal method block. Derived classes of the abstract class must implement all abstract methods. When an abstract class inherits a virtual method from a base class, the abstract class can override the virtual method with an abstract method.

```
//abstract class cannot be instantiated. Provide definiton for subclass (derive)
public abstract class UserAttachment
{
        public int ID { get; private set; }
        public int UserID { get; private set; }
        public string URL { get; protected set; }
    …....

public class GpDocument : UserAttachment   //inherited from UserAttachment class
{
        // delegate, set GetAttachmentTypeId() propertie value, override main class
        public override int GetAttachmentTypeId() => 145;

>> equivalent to <<
>>  public override int getattachmenttypeid{
>>      get { return 145; }
>>    }

        protected GpDocument()   //sub class can use
        { //  }

        //set constructor value from main class constructor
        public GpDocument(int userID) : this(userID, null)
        { }

        //Main class build first, and use this constructor
        public GpDocument(int userID, string url) : base(userID, url)
        {  }

}
```