

UQP - Microsoft Visual Studio

File Edit View Project Build Debug Team Tools Test Analyze Window Help

192.61.99.225

AuditController.cs ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACOMeasureReport.cs

UQP

```
1  using System;
2  using System.Collections.Generic;
3  using System.Data.Entity.Core.Objects;
4  using System.Linq;
5  using System.Net;
6  using System.Net.Http;
7  using System.Web;
8  using System.Web.Http;
9  using UQP.Models;
10 using UQP.Utility;
11 
12 namespace UQP.Controllers
13 {
14     /// <summary>
15     /// This controller handles the Auditor process.
16     /// </summary>
17     [Authorize]
18     public class AuditController : ApiController
19     {
20         // private UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault());
21         //private UQPEntities db = new UQPEntities();
22         static int finalcount;
23 }
```

itController.cs ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACOMeasureReport.cs

```
UQP.UQP.Controllers.AuditController finalcount
21 static int finalcount;
22 [HttpGet]
23 [Route("api/audit/submissionreasons/{id}")]
24 public HttpResponseMessage SubmissionReason(int id)
25 {
26     try
27     {
28         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
29         {
30             List<DropDownValues> ValuesList = new List<DropDownValues>();
31             var query = (from d in db.CustomDropDownLists.AsNoTracking()
32                         where d.Code == id && d.Status == true
33                         select new
34                         {
35                             d.id,
36                             d.Name
37                         }).ToList();
38             if (query.Count != 0)
39             {
40                 foreach (var q in query)
41                 {
42                     DropDownValues value = new DropDownValues
43                     {
44                         id = q.id,
45                         name = q.Name
46                     };
47                     ValuesList.Add(value);
48                 }
49             }
50         }
51     }
52     return Request.CreateResponse(HttpStatusCode.OK, ValuesList);
53 }
```

Output
Show output from: Source Control - Team Foundation

Reconnected from source control because the server is unavailable. To attempt to reconnect to source control, close and then re-open this window.

UQP - Microsoft Visual Studio

192.61.99.225

Edit View Project Build Debug Team Tools Test Analyze Window Help

Debug Any CPU IIS Express (Google Chrome)

AuditController.cs ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACO.MeasureResult.cs

UQP

```
38     if (query.Count != 0)
39     {
40         foreach (var q in query)
41         {
42             DropDownValues value = new DropDownValues
43             {
44                 id = q.id,
45                 name = q.Name
46             };
47             ValuesList.Add(value);
48         }
49     }
50     return Request.CreateResponse(HttpStatusCode.OK, ValuesList);
51 }
52 }
53 }
54 catch (Exception e)
55 {
56     return Request.CreateResponse(HttpStatusCode.BadRequest, e);
57 }
58 }
```

A screenshot of the Microsoft Visual Studio IDE. The menu bar includes 'File', 'Project', 'Build', 'Debug' (set to 'Any CPU'), 'Team', 'Tools', 'Test', 'Analyze', 'Window', and 'Help'. The status bar at the bottom shows the IP address '192.161.09.225'. The code editor displays a C# file named 'UQP.Controllers.AuditController.cs'. The code implements a POST method for an API endpoint '/api/members/gapdetailsaudit/'. It uses Entity Framework to query a database named 'UQPEntities'. The code filters results by provider ID, date range, and status, then paginates them. A dynamic query is used to handle the filtering logic based on provided parameters.

```
[HttpPost]
[Route("api/members/gapdetailsaudit/")]
public HttpResponseMessage gapdetailsaudit([FromUri]PagingParameterModel pagingparametermodel, [FromUri] Searchmember obj, int[] ids)
{
    try
    {
        using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
        {
            String arrTostr = ids.Select(a => a.ToString()).Aggregate((i, j) => i + "," + j);

            int? providerid = Convert.ToInt32(obj.providerid);

            db.Database.CommandTimeout = 240;
            ObjectParameter totalmembercount = new ObjectParameter("totalmembercount", typeof(Int32));
            List<ExtendedGapsSubmitted> lstgap = new List<ExtendedGapsSubmitted>();
            dynamic query = null;

            if (obj.status == "all")
            {
                obj.status = null;
            }
            if (obj.report == true)
            {
                obj.ispagination = false;
                query = db.GapListing(obj.enddate, providerid, obj.status, obj.membername, obj.subsid, arrTostr, obj.healthplanid,
                    obj.docsubmittedby, pagingparametermodel.pageNumber, pagingparametermodel.pageSize, obj.ispagination,
                    obj.auditorid, obj.dob, obj.roleid, totalmembercount).ToList();
            }
        }
    }
}
```

```
itroller.cs X ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACOMeasureReport.cs
UQP.Controllers.AuditController gapdetailsaudit(PagingParameterModel pagingparametermodel, Search
}
else
{
    obj.ispagination = true;
    query = db.GapListing(obj.enddate, providerid, obj.status, obj.membername, obj.subsid, arrfstr,
        obj.healthplanid, obj.docsubmittedby, pagingparametermodel.pageNumber, pagingparametermodel.pageSize,
        obj.ispagination, obj.auditorid, obj.dob, obj.roleid, totalmembercount).ToList();
    finalcount = Convert.ToInt32(totalmembercount.Value);
}

foreach (var per in query)
{
    ExtendedGapsSubmitted objgap = new ExtendedGapsSubmitted();

    objgap.Subscriber_FirstName = per.Subscriber_FirstName;
    //objgap.Subscriber_MidName = per.Subscriber_MidName;
    objgap.Subscriber_LastName = (per.Subscriber_LastName == null && per.Subscriber_FirstName == null) ? "NA" : per.Subscriber_LastName;
    objgap.PROVIDER_FirstName = per.PROVIDER_FirstName;
    //objgap.PROVIDER_MidName = per.PROVIDER_MidName;
    objgap.PROVIDER_LastName = (per.PROVIDER_LastName == null && per.PROVIDER_FirstName == null) ? "NA" : per.PROVIDER_LastName;
    objgap.HealthPlanName = per.HealthPlanName;
    objgap.Measure_Name = per.Measure_Name;
    objgap.DateOfBirth = per.DateofBirth;
    objgap.Gender = per.Gender;
    objgap.status = per.status;
    objgap.SubscriberID = per.SubscriberID;

    objgap.Gapsid = per.Gapsid;
}

Output from: Source Control - Team Foundation
active solution has been temporarily disconnected from source control because the server is unavailable. To attempt to reconnect to source control, close and then re-open the
```

Debug Any CPU IIS Express (Google Chrome)

```
roller.cs ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACOMeasureReport.cs
UQP.Controllers.AuditController gapdetailsaudit(PagingParameterModel pagingparametmodel, S
```

```
objgap.Status = per.Status;
objgap.SubscriberID = per.SubscriberID;

objgap.Gapsid = per.Gapsid;

objgap.ProviderID = per.ProviderID;
objgap.HealthPlanID = per.HealthPlanID;
objgap.MeasureID = per.MeasureID;
objgap.auditdate = per.auditdate;
objgap.Auditorname = per.Auditorname;
objgap.submissiondate = per.submissiondate;
objgap.createdfirstname = per.createdfirstname;
objgap.createdlastname = per.createdlastname;
objgap.reason =(per.status== "rejected") ? per.Value:"";
if(per.status == "rejected" && (objgap.reason==null||objgap.reason==""))
{
    objgap.reason = "No rejected reason has been submitted by Quality Audit";
}
objgap.comment = per.comment;
objgap.DOS = per.DOS;
objgap.ccc_result = per.ccc_result;
objgap.ResultValue = per.ResultValue;
objgap.dee_result = per.dee_result;
lstgap.Add(objgap);

}

if (obj.report == false)
{
```

```
16 if (obj.report == false)
17 {
18     #region pagination
19     // Get's No of Rows Count
20     int count = finalcount;
21     //int count = lstmembermart.Count();
22
23     // Parameter is passed from Query string if it is null then it default Value will be pageNumber:1
24     int CurrentPage = pagingparametermodel.pageNumber;
25
26     // Parameter is passed from Query string if it is null then it default Value will be pageSize:20
27     int PageSize = pagingparametermodel.pageSize;
28
29     // Display TotalCount to Records to User
30     int TotalCount = count;
31
32     // Calculating Totalpage by Dividing (No of Records / Pagesize)
33     int TotalPages = (int)Math.Ceiling(count / (double)PageSize);
34
35     // Returns List of Customer after applying Paging
36     var items = lstgap/*.Skip((CurrentPage - 1) * PageSize).Take(PageSize).ToList()*/;
37
38     // if CurrentPage is greater than 1 means it has previousPage
39     var previousPage = CurrentPage > 1 ? "Yes" : "No";
40
41     // if TotalPages is greater than CurrentPage means it has nextPage
42     var nextPage = CurrentPage < TotalPages ? "Yes" : "No";
43
44     // Object which we are going to send in header
45     var paginationMetadata = new
46     {
47         totalCount = TotalCount,
```

Output from: Source Control - Team Foundation

The active solution has been temporarily disconnected from source control because the server is unavailable. To attempt to reconnect to source control, close and then re-open the solution.

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Title Bar:** Shows multiple open files: UQP.cs, ACUComments.cs, ACO_MembersMarts.cs, BulkEmailController.cs, and ACO_Report.cs.
- Code Editor:** Displays the `UQP.Controllers.AuditController` class. The code is written in C# and handles report generation based on a boolean parameter `obj.report`. It uses a `ReportsController` object to generate reports and returns the response.
- Status Bar:** Shows "100 % 4" and "Output".
- Output Window:** Displays a message about source control disconnection: "The active solution has been temporarily disconnected from source control because the server is unavailable. To attempt to reconnect to source control, close this window and try again."

```
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220

if (obj.report == false)
{
    pagination
}

else if (obj.report == true)
{

    var rprobj = new ReportsController
    {
        Request = new System.Net.Http.HttpRequestMessage(),
        Configuration = new HttpConfiguration()
    };

    return rprobj.GenerateReport(null,
        null, null, "Quality Report", obj.type, obj, null; lstgap,null);
    //return Request.CreateResponse(HttpStatusCode.OK, "ok");
}

return Request.CreateResponse(HttpStatusCode.OK, lstgap);
}

}
catch (Exception ex)
{
    return Request.CreateResponse(HttpStatusCode.InternalServerError, ex.Message);
}
```

```
HubController.cs X ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACOMeasureReport.cs
UQP UQP.Controllers.AuditController gapdetailsaudit(PagingParameterModel pagingparametermodel, Search...
```

219 }

220

221

222 [HttpPost]

223 [Route("api/members/updateauditstatus/{gapsid}")]
 References

224 public HttpResponseMessage updateauditstatus(int gapsid, GapInputSubmit objgaps)

225 {

226 try

227 {

228 using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))

229 {

230 var entity = db.GapsSubmitteds.FirstOrDefault(e => e.Gapsid == gapsid);

231

232 var recheck = (from g in db.GapsSubmitteds

233 where g.MeasureID == entity.MeasureID && g.ipa_id == entity.ipa_id && g.SubscriberID == entity.SubscriberID

234 && (g.Gapsid > entity.Gapsid).

235 select new

236 {

237 g.createdfirstname,

238 g.createdlastname,

239 g.status,

240 g.Measure_Name,

241 g.MeasureID,

242 g.Gapsid

243 }

244).OrderByDescending(e => e.Gapsid).FirstOrDefault();

245

246

247 if (entity != null && recheck == null)

248 {

100 % 4

Output

Show output from: Source Control - Team Foundation

The active solution has been temporarily disconnected from source control because the server is unavailable. To attempt to reconnect to source control, close and then reopen the solution.

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Title Bar:** Debug, Any CPU, IIS Express (Google Chrome).
- Solution Explorer:** Shows files like ProviderController.cs, UQP.Controllers.AuditController, ACOComment.cs, ACO_MemberMart.cs, BulkEmailController.cs, and ACOMeetingController.
- Code Editor:** Displays the content of the AuditController.cs file. The code handles updating an entity and saving changes, including logic for audit status and reason, and creating a new Gapcomment entry if a comment is provided.

```
if (entity != null && recheck == null)
{
    entity.status = objgaps.status;
    // entity.auditdate = DateTime.Now;
    if (objgaps.userid != 0 && (objgaps.rolename != "Administrator" && objgaps.rolename != "Executive Management"))
    {
        entity.Auditorname = objgaps.username;
        entity.auditordid = Convert.ToInt32(objgaps.userid);
        entity.auditdate = DateTime.Now;
    }
    if (objgaps.rolename == "Administrator" || objgaps.rolename == "Executive Management")
    {
        entity.Byadmin = true;
    }
    if (objgaps.reasonid != 0)
    {
        entity.PrimaryReasonID = objgaps.reasonid;
    }
    db.SaveChanges();

    if (objgaps.comment != null)
    {
        Gapcomment comment = new Gapcomment
        {
            userid = objgaps.userid,
            gapid = gapsid,
            createdate = DateTime.Now,
            comment = objgaps.comment,
            username = objgaps.username,
            rolename = objgaps.rolename
        };
    }
}
```

microsoft Visual Studio

File Project Build Debug Team Tools Test Analyze Window Help

192.61.99.225

UQP.Controllers.AuditController

```
    comment = objgaps.comment,
    username = objgaps.username,
    rolename = objgaps.rolename
};

db.Gapcomments.Add(comment);
db.SaveChanges();
}
return Request.CreateResponse(HttpStatusCode.OK, "Status has been updated successfully");
}
else if (recheck != null)
{
    return Request.CreateResponse(HttpStatusCode.BadRequest, "Status of this gap cannot be reversed as Quality Assurance : " +
        recheck.createdlastname + " " + recheck.createdfirstname + " already worked on it");
}
else
{
    return Request.CreateResponse(HttpStatusCode.OK, "No data found for the given gapid" + gapsid);
}

}

catch (Exception ex)
{
    return Request.CreateResponse(HttpStatusCode.InternalServerError, ex.Message);
}
```

```
ProvidersController.cs      Source Control Explorer      ACOComment.cs      ACO_MembersMart.cs      BulkEmailController.cs      ACOMeasureReport.cs
UQP.Controllers.AuditController      Gapcorrection(GapInputSubmit gapInput)

}

//Only executive can approve rejected gap and vice versa
[HttpPost]
[Route("api/members/gapcorrection")]
References
public HttpResponseMessage Gapcorrection(GapInputSubmit gapInput)
{
    try
    {
        using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
        {
            var query = (from g in db.GapsSubmitteds
                        where g.MeasureID == gapInput.measureid && g.SubscriberID.StartsWith(gapInput.SubscriberID)
                        && g.ipa_id == gapInput.ipaid && g.status != "rejected" && g.Gapsid != gapInput.gapsid
                        select new
                        {
                            g.createdfirstname,
                            g.createdlastname,
                            g.Auditorname,
                            g.submissiondate,
                            g.Gapsid
                        }).OrderByDescending(e => e.Gapsid).FirstOrDefault();
            if (query != null)
            {
                return Request.CreateResponse(HttpStatusCode.OK, "Status of " + gapInput.SubscriberID + " cannot be changed as " + query.createdlastname
                + " " + query.createdfirstname + " is already working on it.");
            }
            else
            {
                return Request.CreateResponse(HttpStatusCode.OK, true);
            }
        }
    }
}
```

Source control is unavailable. To attempt to reconnect to source control, close and then re-open the solution.

```
322         g.submissiondate,
323         g.Gapsid
324     }).OrderByDescending(e => e.Gapsid).FirstOrDefault();
325 
326     if (query != null)
327     {
328         return Request.CreateResponse(HttpStatusCode.OK, "Status of " + gapInput.SubscriberID + " cannot be changed as " +
329     }
330     else
331     {
332         return Request.CreateResponse(HttpStatusCode.OK, true);
333     }
334 }
335 catch (Exception ex)
336 {
337     return Request.CreateResponse(HttpStatusCode.OK, ex.InnerException);
338 }
339 }
340
341 // [AllowAnonymous]
342 // [HttpGet]
343 // [Route("api/members/getipaddress")]
344 // public string GetIPAddress()
345 //{
346 //     System.Web.HttpContext context = System.Web.HttpContext.Current;
347 //     string ipAddress = context.Request.ServerVariables["HTTP_X_FORWARDED_FOR"];
348 //
349 //     if (!string.IsNullOrEmpty(ipAddress))
350 //     {
351 //         ----->----->----->----->
```

100 %

Output

Show output from: Source Control - Team Foundation

The active solution has been temporarily disconnected from source control because the server is unavailable. To attempt to reconnect to source control,

```
370
371     [HttpPost]
372     [Route("api/audit/MRAAudit/")]
373     public HttpResponseMessage MRASuspected(Rolewise_users roleobj)
374     {
375
376         try
377         {
378             using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
379             {
380                 int[] abc = roleobj.uniqueids.Select(x => x.id).ToArray();
381                 var condition = db.MRAReviews.Where(x => abc.Contains(x.id));
382
383
384                 var splitted = roleobj.firstname.Split(new[] { ' ' }, 2);
385                 string createdbyfirstname = splitted[0];
386                 string createdbylastname = splitted[1];
387
388
389                 foreach (var per in condition.ToList())
390                 {
391                     foreach (var stats in roleobj.uniqueids.ToList())
392                     {
393                         if (stats.id == per.id)
394                         {
395
396                             per.auditstatus = stats.auditstatus;
397                             per.StatusReason = stats.StatusReason;
398
399                         if (stats.StatusReason == "Provider does not agree")
400                         {
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
```

```
roller.cs ProvidersController.cs Source Control Explorer ACOComment.cs ACO_MembersMart.cs BulkEmailController.cs ACOMeasureReport.cs
UQP.Controllers.AuditController Gapcorrection(GapInputSubmit gapInput)

        if (stats.StatusReason == "Provider does not agree")
        {
            var doc =
                db.MRAReviewDocuments.Where(x => x.MRAId == stats.uniqueid);
            doc.ToList().ForEach(c => c.status = false);

        }

    }

    if (roleobj.uniqueids.Select(x => x.StatusReason).Contains("Not Documented"))
    {
        per.followup = per.followup.Value.AddDays(30);
    }

    per.audittedby = Convert.ToInt32(roleobj.userid);
    per.auditorfirstname = createdbyfirstname;
    per.auditorlastname = createdbylastname;
    per.auditedDate = DateTime.Now;

}

db.SaveChanges();
```

```
423     per.auditorlastname = createdby.lastname;
424     per.auditdate = DateTime.Now;
425
426     }
427
428     db.SaveChanges();
429
430
431
432
433
434     return Request.CreateResponse(HttpStatusCode.OK, "status updated");
435 }
436
437 }
438 catch (Exception e)
439 {
440     return Request.CreateResponse(e.InnerException);
441 }
442 }
443
444 }
445
446 }
```