

```
QP
```

```
1142
1143
1144
1145
1146
1147 [HttpGet]
1148 [Route("api/ACO/measures")]
1149 Preferences
1150 public HttpResponseMessage ACOMeasures()
1151 {
1152     try
1153     {
1154         List<ACO_Measure> measurelist = new List<ACO_Measure>();
1155         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
1156         {
1157             var query = (from m in db.ACO_Measures
1158                         select new
1159                         {
1160                             m.Measure_ID,
1161                             m.Measure_Name
1162                         }).Distinct().ToList();
1163             foreach (var value in query)
1164             {
1165                 ACO_Measure measure = new ACO_Measure
1166                 {
1167                     measureid = Convert.ToInt64(value.Measure_ID),
1168                     measurename = value.Measure_Name
1169                 };
1170                 measurelist.Add(measure);
1171             }
1172             return Request.CreateResponse(HttpStatusCode.OK, measurelist);
1173         }
1174     catch (Exception ex)
1175     {
1176         return Request.CreateResponse(HttpStatusCode.InternalServerError, ex);
1177     }
1178 }
```

```
78
79
80
81
82 [HttpPost]
83 [Route("api/ACO/scorecard")]
84 References
85 public HttpResponseMessage ACOScorecard(ACO_Filter filter)
86 {
87     try
88     {
89         List<MeasureCalculations> measurelist = new List<MeasureCalculations>();
90         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
91         {
92             filter.startdate = filter.startdate.Value;
93             var fromyear1 = filter.startdate.Value.Year;
94
95             int[] ids1 = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
96
97             var query = db.ACOTotalReports.Where(e => (e.CalculatedMonth == filter.startdate) &&
98             (filter.groupid == null || e.GroupID == filter.groupid)
99             && (filter.npi == "" || filter.npi == null || e.ProviderNPI == filter.npi) &&
100            (ids1.Contains((int)e.MeasureId) && e.FromYear.Year == fromyear1)
101            ).GroupBy(e => e.MeasureId).Select(result =>
102                new
103                {
104                    Measureid = result.Select(y => y.MeasureId).FirstOrDefault(),
105                    Measurename = result.Select(y => y.MeasureName).FirstOrDefault(),
106                    Numerator = result.Select(y => y.Numerator).Sum(),
107                    Denominator = result.Select(x => x.FinalDenominator).Sum(),
108                    Gap = result.Select(x => x.gap).Sum(),
109                    Exclusion = result.Select(x => x.Exclusion).Sum(),
110                    // Percentage = result.Select(x => x.PercentCompleted).Average(),
111                }).ToList();
112     }
113 }

```

```

    // Percentage = result.Select(x => x.PercentCompleted).Average(),
    }).ToList();
}

var pendingcount = (from e in db.ACO_Gapsubmitted
join
fd in db.ACO_Measures_Members
on new { sid = e.subscriberid, mid = (int?)e.measureid } equals new { sid = fd.BENE_HIC_NUM, mid = (int?)fd.MeasureId }
where (e.submissiondate<= filter.enddate) &&
(filter.groupid == null || fd.Group_ID == filter.groupid)
&& (filter.npi == "" || filter.npi == null || fd.ProviderNPI == filter.npi) &&
(ids1.Contains((int)fd.MeasureId) && fd.fromyear.Value.Year == fromyear1
)
&& fd.documentstatus == "pending"
group new { e, fd } by new
{
    fd.MeasureId
} into gr
select new
{
    measureid = gr.Select(x => x.fd.MeasureId).FirstOrDefault(),
    count = gr.Select(y => y.fd.BENE_HIC_NUM).Distinct().Count()
}).ToList();

if (query.Count() != 0)
{
    foreach (var value in query)
    {
        measureCalculations measure = new measureCalculations
        {
            measureid = value.Measureid,
            measurename = value.Measurename,
        };
    }
}

```

```
    measureid = value.Measureid,
    measurename = value.Measurename,
    num = value.Numerator,
    den = value.Denominator,
    gaps = value.Gap,
    exclusion = value.Exclusion,
    completed = (value.Denominator == 0) ? 0 : (Convert.ToDouble(value.Numerator) * 100 / Convert.ToDouble(value.Denominator)),
    totalpending = pendingcount.Where(x => x.measureid == value.Measureid).Select(x => x.count).FirstOrDefault()
}
measurelist.Add(measure);
}
}
if (filter.report == true)
{
    ScoreCardReportInput obj = new ScoreCardReportInput();
    obj.providername = filter.providername;
    obj.groupname = filter.groupname;
    obj.aconame = "Integral";
    obj.year = (filter.startdate != null) ? filter.startdate.Value.Year.ToString() : null;
    obj.selectedmonth = (filter.startdate != null) ? filter.startdate.Value.ToString("MM") : null;

    var rpoj = new ReportsController
    {
        Request = new System.Net.Http.HttpRequestMessage(),
        Configuration = new HttpConfiguration()
    };
    return rpoj.GenerateReport(obj, null, null, "ACO Scorecard", "excel", null, null, null, measurelist);
}
else
{
    if (filter.name == "lowestcompliance")
}
```

JQP - 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)

UQPController.cs ProvidersController.cs Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs

UQP.Controllers.ACOController finalcount

```
1276     else
1277     {
1278         if (filter.name == "lowestcompliance")
1279         {
1280             return Request.CreateResponse(HttpStatusCode.OK, measurelist.OrderBy(x=>x.completed));
1281         }
1282         else
1283         {
1284             return Request.CreateResponse(HttpStatusCode.OK, measurelist.OrderBy(x => x.measurename));
1285         }
1286     }
1287     catch (Exception ex)
1288     {
1289         return Request.CreateResponse(HttpStatusCode.InternalServerError, ex);
1290     }
1291 }
```

Debug - Any CPU IIS Express (Google Chrome) ProvidersController.cs ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs

UQP Controllers.ACOController finalcount

```
1294 //<summary>
1295 //<param name="gaps"></param>
1296 //<returns></returns>
1297 [HttpPost]
1298 [Route("api/ACO/gapsSubmission")]
1299 References
1300 public HttpResponseMessage GapsSubmission(ACO_Gaps gaps)
1301 {
1302     try
1303     {
1304         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
1305         {
1306             var firstSpaceIndex = gaps.submittedfirstname.IndexOf(" ");
1307             var firstname = gaps.submittedfirstname.Substring(0, firstSpaceIndex);
1308             var lastname = gaps.submittedfirstname.Substring(firstSpaceIndex + 1);
1309             //Below variable is used to store the from year in database from new record
1310             var fromyear = new DateTime(DateTime.Now.Year, 1, 1);
1311             //Below variable is used to store the to year in database from new record
1312             var toyear = new DateTime(DateTime.Now.Year, 12, 31);
1313             string measurename = db.ACO_Measures.Where(x => x.Measure_ID == gaps.measureid).Select(x => x.Measure_Name).FirstOrDefault();
1314
1315             ACO_GapsSubmitted submitted = new ACO_GapsSubmitted();
1316             gaps.measurename = measurename;
1317             submitted.subscriberid = gaps.hicn;
1318             submitted.subscriberfirstname = gaps.subscriberfirstname;
1319             submitted.subscriberlastname = gaps.subscriberlastname;
1320             submitted.measureid = gaps.measureid;
1321             submitted.submissiondate = DateTime.Today;
1322             submitted.submittedby = gaps.submittedby;
1323             submitted.submittedfirstname = firstname;
1324             submitted.submittedlastname = lastname;
1325             submitted.providerfirstname = gaps.providerfirstname;
1326             submitted.providerlastname = gaps.providerlastname;
1327             submitted.npi = gaps.providernpi;
1328         }
1329     }
1330 }
```

```
1326 submitted.providerfirstname = gaps.providerfirstname;
1327 submitted.providerlastname = gaps.providerlastname;
1328 submitted.npi = gaps.providernpi;
1329 submitted.auditby = gaps.auditby;
1330 submitted.auditdate = gaps.auditdate;
1331 submitted.auditfirstname = gaps.auditfirstname;
1332 submitted.auditlastname = gaps.auditlastname;
1333 submitted.othercmsreason = gaps.othercmsreason;
1334 submitted.medicalrecordfound = gaps.medicalrecordfound;
1335 submitted.care2_qualifiedforsample = gaps.care2_qualifiedforsample;
1336 submitted.care2_qualifiedforsamplereason = gaps.care2_qualifiedforsamplereason;
1337 submitted.care2_qualifiedformeasure = gaps.care2_qualifiedformeasure;
1338 submitted.care2_qualifiedformeasurerreason = gaps.care2_qualifiedformeasurerreason;
1339 submitted.care2_patientscreened = gaps.care2_patientscreened;
1340 submitted.dm2_patientdocumentedhistory = gaps.dm2_patientdocumentedhistory;
1341 submitted.dm2_HbA1Cperformed = gaps.dm2_HbA1Cperformed;
1342 submitted.dm2_hb1cdate = gaps.dm2_hb1cdate;
1343 submitted.bcs_qualifiedforsample = gaps.bcs_qualifiedforsample;
1344 submitted.bcs_qualifiedforsamplereason = gaps.bcs_qualifiedforsamplereason;
1345 submitted.bcs_mamogramcompleted = gaps.bcs_mamogramcompleted;
1346 submitted.cc_qualifiedforsample = gaps.cc_qualifiedforsample;
1347 submitted.cc_qualifiedforsamplereason = gaps.cc_qualifiedforsamplereason;
1348 submitted.cc_qualifiedforsamplereason2 = gaps.cc_qualifiedforsamplereason2;
1349 submitted.cc_currentscreening = gaps.cc_currentscreening;
1350 submitted.flu_qualifiedforsample = gaps.flu_qualifiedforsample;
1351 submitted.flu_receivedflushot = gaps.flu_receivedflushot;
1352 submitted.htn_hypertensiondiagnosis = gaps.htn_hypertensiondiagnosis;
1353 submitted.htn_hypertensiondiagnosisisreason = gaps.htn_hypertensiondiagnosisisreason;
1354 submitted.htn_bpdocumented = gaps.htn_bpdocumented;
1355 submitted.htn_bpdate = gaps.htn_bpdate;
1356 submitted.htn_bpssystolic = gaps.htn_bpssystolic;
1357 submitted.htn_bpdytolic = gaps.htn_bpdytolic;
1358 submitted.mh1_majordepressiondiagnosis = gaps.mh1_majordepressiondiagnosis;
1359 submitted.mh1_majordepressiondiagnosisisreason = gaps.mh1_majordepressiondiagnosisisreason;
1360 submitted.mh1_phq9completed = gaps.mh1_phq9completed;
1361 submitted.mh1_phq9greater = gaps.mh1_phq9greater;
```

```
3 submitted.htn_hypertensiondiagnosisreason = gaps.htn_hypertensiondiagnosisreason;
4 submitted.htn_bpdocumented = gaps.htn_bpdocumented;
5 submitted.htn_bpdate = gaps.htn_bpdate;
6 submitted.htn_bpystolic = gaps.htn_bpystolic;
7 submitted.htn_bpdystolic = gaps.htn_bpdystolic;
8 submitted.mhl_majordepressiondiagnosis = gaps.mhl_majordepressiondiagnosis;
9 submitted.mhl_majordepressiondiagnosisreason = gaps.mhl_majordepressiondiagnosisreason;
10 submitted.mhl_phq9completed = gaps.mhl_phq9completed;
11 submitted.mhl_phq9greater = gaps.mhl_phq9greater;
12 submitted.mhl_phq9greaterdate = gaps.mhl_phq9greaterdate;
13 submitted.mhl_phq9gatherscore = gaps.mhl_phq9gatherscore;
14 submitted.mhl_phq9measurementperiodcompleted = gaps.mhl_phq9measurementperiodcompleted;
15 submitted.mhl_phq9measurementperiodcompleteddate = gaps.mhl_phq9measurementperiodcompleteddate;
16 submitted.mhl_phq9less = gaps.mhl_phq9less;
17 submitted.mhl_phq9lessdate = gaps.mhl_phq9lessdate;
18 submitted.mhl_phq9lessscore = gaps.mhl_phq9lessscore;
19 submitted.prev12_qualified = gaps.prev12_qualified;
20 submitted.prev12_depressionscreened = gaps.prev12_depressionscreened;
21 submitted.prev12_positivefordepression = gaps.prev12_positivefordepression;
22 submitted.prev12_followup = gaps.prev12_followup;
23 submitted.tobacco_qualified = gaps.tobacco_qualified;
24 submitted.tobacco_screened = gaps.tobacco_screened;
25 submitted.tobacco_recentscreening = gaps.tobacco_recentscreening;
26 submitted.tobacco_cessatinintervention = gaps.tobacco_cessatinintervention;
27 // submitted.comment = gaps.comment;
28 submitted.groupid = gaps.groupid;
29 submitted.prev13_qualifiedsample = gaps.prev13_qualifiedsample;
30 submitted.prev13_statintreatment = gaps.prev13_statintreatment;
31 submitted.prev13_medreason = gaps.prev13_medreason;
32 submitted.prev13_statintherapy = gaps.prev13_statintherapy;
33 submitted.prev13_LDLC = gaps.prev13_LDLC;
34 submitted.subscribebergender = gaps.gender;
35 submitted.dob = gaps.dob;
```

```
//Exclusion Condition
if (
//care 2
gaps.care2_qualifiedformeasure == "No" ||
//dm2
gaps.dm2_patientdocumentedhistory == "Not Confirmed - Diagnosis" || gaps.dm2_patientdocumentedhistory == "No - Other cons Approved Reason" ||
//htn
gaps.htn_hypertensiondiagnosis == "Not confirmed Diagnosis" || gaps.htn_hypertensiondiagnosis == "Denominator Exclusion" ||
gaps.htn_hypertensiondiagnosis == "No other CMS Approved Reason" ||
//mental health
gaps.mhl_majordepressiondiagnosis == "No - Not confirmed Diagnosis" || gaps.mhl_majordepressiondiagnosis == "Denominator Exclusion" ||
gaps.mhl_phq9completed == "No" || gaps.mhl_phq9greater == "No" ||
//breast cancer
gaps.bcs_qualifiedforsample == "No" || gaps.bcs_qualifiedforsample == "No other CMS approved" ||
//colectral cancer
gaps.cc_qualifiedforsample == "No" || gaps.cc_qualifiedforsample == "No - other CMS reason approved" ||
//flu shot
gaps.flu_qualifiedforsample == "No" || gaps.flu_receivedflushot == "Denominator Exception-Medical reasons" ||
gaps.flu_receivedflushot == "Denominator Exception-Patient reasons" || gaps.flu_receivedflushot == "Denominator Exception-System reasons" ||
//tobacco
gaps.tobacco_qualified == "No" || gaps.tobacco_screened == "No" || gaps.tobacco_screened == "Denominator Exception-Medical reasons" ||
gaps.tobacco_recentscreening == "No" || gaps.tobacco_cessatinintervention == "Denominator Exclusion-Medical reasons" ||
//prev12
gaps.prev12_qualified == "No" || gaps.prev12_qualified == "Denominator Exclusion-depression" ||
gaps.prev12_qualified == "Denominator Exclusion-Bipolar Disorder" || gaps.prev12_depressionscreened == "Denominator Exception-Medical reasons" ||
gaps.prev12_depressionscreened == "Denominator Exception-Patient reasons" ||
|| (gaps.prev13_qualifiedsample!=null &
|| (gaps.prev13_qualifiedsample.Contains("No"))||
gaps.prev13_statintreatment=="No - denominator exception" || gaps.prev13_LDLC=="No" ||
gaps.prev13_statintherapy=="No - denominator exception")
{
    //check if exclusion exist in the table
    var query = (from f in db.ACO_Measures_Members where f.MeasureId == gaps.measureid && f.BENE_HIC_NUM == gaps.hicn select f).ToList();
    if (query.Count() != 0)
```

```
413
414
415 //check if exclusion exist in the table
416 var query = (from f in db.ACO_Measures_Members where f.MeasureId == gaps.measureid && f.BENE_HIC_NUM == gaps.hicn select f).ToList();
417 if (query.Count() != 0)
418 {
419     foreach (var value in query)
420     {
421         value.code_type = "E";
422         value.documentstatus = "exclusion";
423         value.Source_TypeUQP = "byupload";
424         db.SaveChanges();
425     }
426     submitted.IspresentInRecord = true;
427 }
428 else
429 {
430     ACO_Measures_Members newinsert = new ACO_Measures_Members
431     {
432         BENE_HIC_NUM = gaps.hicn,
433         MeasureId = gaps.measureid,
434         MeasureName = gaps.measurename,
435         SubscriberFirstName = gaps.subscriberfirstname,
436         SubscriberLastName = gaps.subscriberlastname,
437         DOB = gaps.dob,
438         Gender = gaps.gender,
439         ProviderFirstName = gaps.providerfirstname,
440         ProviderLastName = gaps.providerlastname,
441         ProviderNPI = gaps.providernpi,
442         code_type = "E",
443         documentstatus = "exclusion",
444         CLM_FROM_DT = DateTime.Today,
445         createDateByUQP = DateTime.Now,
446         fromyear = fromyear,
447         toyear=toyear,
448         Source_TypeUQP = "byupload",
449     };
450     db.ACO_Measures_Members.Add(newinsert);
451 }
```

```
UQP ProvidersController.cs Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs
UQP Controllers\ACOController.cs GapsSubmission(ACO_Gaps gaps)
1446     Source_TypeUQP = "byupload",
1447 };
1448     db.ACO_Measures_Members.Add(newinsert);
1449     db.SaveChanges();
1450     submitted.IsPresentInRecord = false;
1451 }
1452 submitted.DocumentStatus = "exclusion";
1453 db.ACO_GapsSubmitted.Add(submitted);
1454 db.SaveChanges();
1455 }
1456 //Denominator Condition OR Gap
1457 else if (
1458     //care2
1459     gaps.care2_patientscreened == "No" ||
1460     //dm2
1461     gaps.dm2_HbA1Cperformed == "No" ||
1462     //htn
1463     gaps.htn_bpdocumented == "No" || gaps.htn_bpSystolic > 139 || gaps.htn_bpDystolic > 80 ||
1464     //mental health
1465     // gaps.mh1_phq9greater == "Yes" ||
1466     gaps.mh1_phq9measurementperiodcompleted == "No" || gaps.mh1_phq9less == "No" ||
1467     //breast cancer
1468     gaps.bcs_mamogramcompleted == "No" || //
1469     //colectral cancer
1470     gaps.cc_currentscreening == "No" ||
1471     //flu shot
1472     gaps.flu_receivedflushot == "No" ||
1473     //tobacco
1474     gaps.tobacco_cessatinintervention == "Denominator Exclusion-Medical reasons" ||
1475     //prev 12
1476     gaps.prev12_followup == "No" || gaps.prev12_depressionscreened == "No"
1477     || gaps.prev13_statintreatment == "No - measure not met" ||
1478     gaps.prev13_statintherapy == "No - measure not met"
1479 )
1480 {
1481     //check if it exist in the table
```

```
79     }
80     {
81         //check if it exist in the table
82         var query = (from f in db.ACO_Measures_Members where f.MeasureId == gaps.measureid && f.BENE_HIC_NUM == gaps.hicn select f).ToList();
83         if (query.Count() != 0)
84         {
85             foreach (var value in query)
86             {
87                 value.code_type = "D";
88                 value.documentstatus = "notsubmitted";
89                 value.Source_TypeUQP = "byupload";
90                 db.SaveChanges();
91             }
92             submitted.IspresentInRecord = true;
93         }
94         else
95         {
96             ACO_Measures_Members newinsert = new ACO_Measures_Members
97             {
98                 BENE_HIC_NUM = gaps.hicn,
99                 MeasureId = gaps.measureid,
100                MeasureName = gaps.measurename,
101                SubscriberFirstName = gaps.subscriberfirstname,
102                SubscriberLastName = gaps.subscriberlastname,
103                DOB = gaps.dob,
104                Gender = gaps.gender,
105                ProviderFirstName = gaps.providerfirstname,
106                ProviderLastName = gaps.providerlastname,
107                ProviderNPI = gaps.providernpi,
108                code_type = "",
109                documentstatus = "notsubmitted",
110                CLM_FROM_DT = DateTime.Today,
111                createDateByUQP = DateTime.Now,
112                fromyear = fromyear,
113                toyear = toyear,
114                Source_TypeUQP = "byupload"
```

```
514
515     Source_TypeUQP = "byupload"
516 }
517     db.ACO_Measures_Members.Add(newinsert);
518     db.SaveChanges();
519     submitted.IspresentInRecord = false;
520 }
521 submitted.documentstatus = "denominator";
522 db.ACO_Gapsubmitted.Add(submitted);
523 db.SaveChanges();
524 }
525 //Pending or Numerator Complaint
526 else if (
527     //breast cancer
528     gaps.bcs_mamogramcompleted == "Yes" ||
529     //care 2
530     gaps.care2_patientscreened == "Yes" ||
531     //colectral cancer
532     gaps.cc_currentscreening == "Yes" ||
533     //dm2
534     gaps.dm2_HbA1Cperformed == "Yes" ||
535     //flu shot
536     gaps.flu_receivedflushot == "Yes" ||
537     //htn
538     gaps.htn_bpsystolic <= 139 && gaps.htn_bpdiastolic <= 80 ||
539     //mental health
540     gaps.mh1_phq9less == "Yes" ||
541     //prev 12
542     gaps.prev12_followup == "Yes" ||
543     gaps.prev12_positivefordepression=="No" ||
544     //tobacco
545     gaps.tobacco_cessatinintervention == "Yes"
```

```
UQP.Controllers.ACOController
```

```
Gaps.mh1_phq9less == "Yes" ||
//prev 12
gaps.prev12_followup == "Yes" ||
gaps.prev12_positivefordepression=="No" ||
//tobacco
gaps.tobacco_cessatinintervention == "Yes"
|| gaps.prev13_statintreatment=="Yes" || gaps.prev13_statintherapy=="Yes"
)
{
    //check if it exist in the table
    var query = (from f in db.ACQ_Measures_Members where f.MeasureId == gaps.measureid && f.BENE_HIC_NUM == gaps.hicn select f).ToList();
    if (query.Count() != 0)
    {
        foreach (var value in query)
        {
            value.code_type = "D";
            value.documentstatus = "pending";
            value.Source_TypeUQP = "byupload";
            db.SaveChanges();
        }
        submitted.IspresentInRecord = true;
    }
    else
    {
        ACQ_Measures_Members newinsert = new ACQ_Measures_Members
        {
            BENE_HIC_NUM = gaps.hicn,
            MeasureId = gaps.measureid,
            MeasureName = gaps.measurename,
            SubscriberFirstName = gaps.subscriberfirstname,
            SubscriberLastName = gaps.subscriberlastname,
            DOB = gaps.dob.Value.Date,
            Gender = gaps.gender,
            ProviderFirstName = gaps.providerfirstname,
            ProviderLastName = gaps.providerlastname,
            ProviderNPI = gaps.providernpi,
        }
    }
}
```

```
ProviderFirstName = gaps.providerfirstname,
ProviderLastName = gaps.providerlastname,
ProviderNPI = gaps.providernpi,
code_type = "D",
documentstatus = "pending",
CLM_FROM_DT = DateTime.Today,
createDateByUQP = DateTime.Now,
fromyear = fromyear,
toyear = toyear,
Source_TypeUQP = "byupload",
};

db.ACO_Measures_Members.Add(newinsert);
db.SaveChanges();
submitted.IspresentInRecord = false;
}
submitted.documentstatus = "pending";
db.ACO_Gapsubmitted.Add(submitted);
db.SaveChanges();
submitted.gapsid = submitted.gapsid;

}

//It will maintain the history and it will also store the comment will run in every condition
ACOComment comment = new ACOComment
{
    userid = submitted.submittedby,
    gapid = submitted.gapsid,
    createdate = DateTime.Now,
    comment = gaps.comment,
    username = lastname + " " + firstname,
    Subscriberid = gaps.hicn,
    Measureid = gaps.measureid,
    Documentstatus = submitted.documentstatus,
    rolename = gaps.rolename,
```

UQP - Microsoft Visual Studio

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

Debug Any CPU IIS Express (Google Chrome)

ACOController.cs ProvidersController.cs Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs GapsSubmission(ACO_Gaps gaps)

UQP

```
1599     userid = submitted.submittedby,
1600     gapid = submitted.gapid,
1601     createdate = DateTime.Now,
1602     comment = gaps.comment,
1603     username = lastname + " " + firstname,
1604     Subscriberid = gaps.hicn,
1605     Measureid = gaps.measureid,
1606     Documentstatus = submitted.documentstatus,
1607     rolename = gaps.rolename,
1608     );
1609     db.ACComments.Add(comment);
1610     db.SaveChanges();
1611
1612     return Request.CreateResponse(HttpStatusCode.OK, submitted);
1613   }
1614
1615   }
1616   catch (Exception ex)
1617   {
1618     return Request.CreateResponse(HttpStatusCode.InternalServerError, ex);
1619   }
1620 }
1621 }
```

```
1623 [HttpPost]
1624 [Route("api/ACO/getmembergap")]
1625 public HttpResponseMessage GetcurrentUsermeasurestatus(ACO_Member member)
1626 {
1627     try
1628     {
1629         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
1630         {
1631             List<ACO_Comment_Docs> comment_docslist = new List<ACO_Comment_Docs>();
1632             dynamic gaps = null;
1633             //user is on aco listing
1634             if (member.gapsid == null)
1635             {
1636                 gaps = db.ACOPsubmitted.Where(e => e.measureid == member.measureid && e.subscriberid == member.hicn)
1637                     .OrderByDescending(e => e.gapsid).FirstOrDefault();
1638             }
1639             //user is on gap listing
1640             else
1641             {
1642                 gaps = db.ACOPsubmitted.Where(e => e.gapsid == member.gapsid).FirstOrDefault();
1643             }
1644             ACO_Gaps submitted = new ACO_Gaps();
1645
1646             if (gaps != null)
1647             {
1648                 if (member.gapsid == null && gaps.documentstatus == "rejected")
1649                 {
1650                     submitted.type = "denominator";
1651                     // When user is on aco listing and open a rejected member, the data should not populate. Therefore it is left intentionally blank
1652                 }
1653             }
1654             else
1655             {
1656                 submitted.sourcetype = "byupload";
1657                 submitted.gapsid = gaps.gapsid;
1658                 submitted.hicn = gaps.subscriberid;
1659             }
1660         }
1661     }
1662 }
```

```
submitted.sourcetype = "byupload";
submitted.gapsid = gaps.gapsid;
submitted.h
long ACO_Gapsubmitted.gapsid { get; set; }
submitted.s
tname;
submitted.subscriberlastname = gaps.subscriberlastname;
submitted.measureid = gaps.measureid;
submitted.submissiondate = gaps.submissiondate;
submitted.submittedby = gaps.submittedby;
submitted.submittedfirstname = gaps.submittedfirstname;
submitted.submittedlastname = gaps.submittedlastname;
submitted.providerfirstname = gaps.providerfirstname;
submitted.providerlastname = gaps.providerlastname;
submitted.providernpi = gaps.npi;
submitted.auditby = gaps.auditby;
submitted.auditdate = gaps.auditdate;
submitted.auditfirstname = gaps.auditfirstname;
submitted.auditlastname = gaps.auditlastname;
submitted.othercmsreason = gaps.othercmsreason;
submitted.medicalrecordfound = gaps.medicalrecordfound;
submitted.care2_qualifiedforsample = gaps.care2_qualifiedforsample;
submitted.care2_qualifiedforsamplereason = gaps.care2_qualifiedforsamplereason;
submitted.care2_qualifiedformeasure = gaps.care2_qualifiedformeasure;
submitted.care2_qualifiedformearureason = gaps.care2_qualifiedformearureason;
submitted.care2_patientscreened = gaps.care2_patientscreened;
submitted.dm2_patientdocumentedhistory = gaps.dm2_patientdocumentedhistory;
submitted.dm2_HbA1Cperformed = gaps.dm2_HbA1Cperformed;
submitted.dm2_hba1cdate = gaps.dm2_hba1cdate;
submitted.bcs_qualifiedforsample = gaps.bcs_qualifiedforsample;
submitted.bcs_qualifiedforsamplereason = gaps.bcs_qualifiedforsamplereason;
submitted.bcs_mamogramcompleted = gaps.bcs_mamogramcompleted;
submitted.cc_qualifiedforsample = gaps.cc_qualifiedforsample;
submitted.cc_qualifiedforsamplereason = gaps.cc_qualifiedforsamplereason;
submitted.cc_qualifiedforsamplereason2 = gaps.cc_qualifiedforsamplereason2;
submitted.cc_currentscreening = gaps.cc_currentscreening;
submitted.flu_qualifiedforsample = gaps.flu_qualifiedforsample;
submitted.flu_receivedflushot = gaps.flu_receivedflushot;
```

Source Control Explorer ACO_MemberServices BulkEmailController.cs Notification_UserSpecified.cs

UQP.Controllers.ACOController GetcurrentUsermeasurestatus(ACO_Member member)

```
submitted.cc_currentscreening = gaps.cc_currentscreening;
submitted.flu_qualifiedforsample = gaps.flu_qualifiedforsample;
submitted.flu_receivedflushot = gaps.flu_receivedflushot;
submitted.htn_hypertensiondiagnosis = gaps.htn_hypertensiondiagnosis;
submitted.htn_hypertensiondiagnosisreason = gaps.htn_hypertensiondiagnosisreason;
submitted.htn_bpdocumented = gaps.htn_bpdocumented;
submitted.htn_bpdate = gaps.htn_bpdate;
submitted.htn_bpystolic = gaps.htn_bpystolic;
submitted.htn_bpystolic = gaps.htn_bpystolic;
submitted.mh1_majordepressiondiagnosis = gaps.mh1_majordepressiondiagnosis;
submitted.mh1_majordepressiondiagnosisreason = gaps.mh1_majordepressiondiagnosisreason;
submitted.mh1_phq9completed = gaps.mh1_phq9completed;
submitted.mh1_phq9greater = gaps.mh1_phq9greater;
submitted.mh1_phq9greaterdate = gaps.mh1_phq9greaterdate;
submitted.mh1_phq9greaterscore = gaps.mh1_phq9greaterscore;
submitted.mh1_phq9measurementperiodcompleted = gaps.mh1_phq9measurementperiodcompleted;
submitted.mh1_phq9measurementperiodcompleteddate = gaps.mh1_phq9measurementperiodcompleteddate;
submitted.mh1_phq9less = gaps.mh1_phq9less;
submitted.mh1_phq9lessdate = gaps.mh1_phq9lessdate;
submitted.mh1_phq9lessscore = gaps.mh1_phq9lessscore;
submitted.prev12_qualified = gaps.prev12_qualified;
submitted.prev12_depressionscreened = gaps.prev12_depressionscreened;
submitted.prev12_positivefordepression = gaps.prev12_positivefordepression;
submitted.prev12_followup = gaps.prev12_followup;
submitted.tobacco_qualified = gaps.tobacco_qualified;
submitted.tobacco_screened = gaps.tobacco_screened;
submitted.tobacco_recentscreening = gaps.tobacco_recentscreening;
submitted.tobacco_cessatinintervention = gaps.tobacco_cessatinintervention;
submitted.documentstatus = gaps.documentstatus;
submitted.prev13_LDLC = gaps.prev13_LDLC;
submitted.prev13_medreason = gaps.prev13_medreason;
submitted.prev13_qualifiedsample = gaps.prev13_qualifiedsample;
submitted.prev13_statintherapy = gaps.prev13_statintherapy;
submitted.prev13_statintreatment = gaps.prev13_statintreatment;
if (gaps.documentstatus == "approved")
{
```

F11 FileSearch Components Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs

UQP.Controllers.ACOController

GetcurrentUsermeasurestatus(ACO_Member member)

```
21 submitted.prev13_statintreatment = gaps.prev13_statintreatment;
22 if (gaps.documentstatus == "approved")
23 {
24     submitted.type = "numerator";
25 }
26 else if (gaps.documentstatus == "rejected")
27 {
28     submitted.type = "denominator";
29 }
30 else
31 {
32     submitted.type = gaps.documentstatus;
33 }
34
35 var docquery = (from d in db.ACODocsubmitteds
36                 join g in db.ACOGapSubmitted on d.ACOGapSubmittedid equals g.gapsid
37
38                 where g.subscriberid == member.hicn && g.measureid == member.measureid
39                 select new
40                 {
41                     d.FileName,
42                     d.FilePath,
43                     d.Docid,
44                     d.ACOGapSubmittedid
45                 }).ToList();
46 List<ACO_Docs> doclist = new List<ACO_Docs>();
47 foreach (var value in docquery)
48 {
49     ACO_Docs docs = new ACO_Docs
50     {
51         filename = value.FileName,
52         filepath = value.FilePath,
53         documentid = value.Docid,
54         gapsid = value.ACOGapSubmittedid.Value
55     };
56     doclist.Add(docs);
57 }
```

Project Build Debug Team Tools Test Analyze Window Help

Debug Any CPU IIS Express (Google Chrome)

COController.cs* ProvidersController.cs Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs

UQP UQP.Controllers.ACOController

```
1754     gapid = value.AC0GapSubmittedid.Value
1755 }
1756 doclist.Add(docs);
1757 }
1758 submitted.docs = doclist;
1759
1760 //var comment = (from c in db.AC0Comments
1761 //                  where c.Subscriberid==member.hicn && c.Measureid==member.measureid
1762 //                  select c);
1763 //((db.CAPHS_HOS_Question.Where(e => e.type == "HOS").GroupJoin
1764 //      ((db.CAPHS_HOS_Answer.DefaultIfEmpty(), p => p.questionid, c => c.questionid, (p, c1) => new { p.question, p.questionid,
1765 //p.mainquestionid, ans = c1.ToList() }))
1766 //      .ToList();
1767
1768 }
1769 var comment = (db.AC0Comments.Where(e => e.Subscriberid == member.hicn && e.Measureid == member.measureid)
1770 .GroupJoin(db.AC0Docsubmitteds.DefaultIfEmpty(), p => p.gapid, c => c.AC0GapSubmittedid, (p, c1) => new
1771 {
1772     p.userid,
1773     p.gapid,
1774     p.createdate,
1775     p.comment,
1776     p.username,
1777     p.Subscriberid,
1778     p.Measureid,
1779     p.Documentstatus,
1780     p.rolename,
1781     docs = c1.ToList()
1782 }).ToList();
1783 if (comment != null)
1784 {
1785     foreach (var value in comment)
1786     {
1787         foreach (var doc in value.docs)
1788         {
1789             if (doc.gapid == gapid)
1790             {
1791                 doc.docstatus = 1;
1792             }
1793         }
1794     }
1795 }
```

```
1784
1785     if (comment != null)
1786     {
1787
1788         foreach (var value in comment)
1789         {
1790
1791             ACO_Comment_Docs c = new ACO_Comment_Docs();
1792             List<ACO_Docs> documentlist = new List<ACO_Docs>();
1793             c.userid = value.userid;
1794             c.gapid = value.gapid;
1795             c.createdate = value.createdate;
1796             c.comment = value.comment;
1797             c.username = value.username;
1798             c.Subscriberid = value.Subscriberid;
1799             c.Measureid = value.Measureid;
1800             c.Documentstatus = value.Documentstatus;
1801             c.rolename = value.rolename;
1802             foreach (var doc in value.docs)
1803             {
1804                 ACO_Docs document = new ACO_Docs
1805                 {
1806                     filename = doc.FileName,
1807                     filepath = doc.FilePath,
1808                     documentid = doc.Docid,
1809                     gapid = doc.AC0Gapsubmittedid.Value
1810                 };
1811                 documentlist.Add(document);
1812             }
1813             c.documentlist = documentlist;
1814             comment_docslist.Add(c);
1815         }
1816         submitted.commentlist = comment_docslist;
1817     }
1818 }
1819 }
```

100 %

Output

Central - Team Foundation



ACOController.cs* > X ProvidersController.cs Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs

UQP UQP.Controllers.ACOController

```
1814     comment_docslist.Add(c);
1815 }
1816 }
1817 }
1818 }
1819 else
1820 {
1821     var query = db.ACO_Measures_Members.Where(e => e.MeasureId == member.measureid && e.BENE_HIC_NUM == member.hicn);
1822     Select(e => new { e.code_type,e.Source_TypeUQP }).FirstOrDefault();
1823
1824     if (query != null)
1825     {
1826         if (query.code_type == "E")
1827         {
1828             submitted.type = "exclusion";
1829         }
1830         else if (query.code_type == "N")
1831         {
1832             submitted.type = "numerator";
1833         }
1834         else if (query.code_type == "D")
1835         {
1836             submitted.type = "denominator";
1837         }
1838         submitted.sourcetype = query.Source_TypeUQP;
1839     }
1840     else
1841     {
1842         submitted.sourcetype = null;
1843     }
1844
1845     if (member.firsttransition == true)
1846     {
1847         member.firsttransition = false;
1848         member.lasttransition = DateTime.Now;
1849     }
1850 }
```

```
1844     submitted.sourcetype = null;
1845
1846     }
1847
1848     if (member.firsttransition == true)
1849     {
1850         List<int> num = new List<int>();
1851         List<int> den = new List<int>();
1852         List<int> exc = new List<int>();
1853         List<int> pen = new List<int>();
1854         var query = (from m in db.ACO_Measures_Members
1855                     where m.BENE_HIC_NUM == member.hicn
1856                     select new
1857                     {
1858                         m.code_type,
1859                         m.documentstatus,
1860                         m.MeasureId
1861                     }).Distinct().ToList();
1862
1863         foreach (var value in query)
1864         {
1865             if (value.code_type == "D" && value.documentstatus != "pending")
1866             {
1867                 den.Add(value.MeasureId.Value);
1868             }
1869             if (value.code_type == "D" && value.documentstatus == "pending")
1870             {
1871                 pen.Add(value.MeasureId.Value);
1872             }
1873             if (value.code_type == "E")
1874             {
1875                 exc.Add(value.MeasureId.Value);
1876             }
1877             if (value.code_type == "I")
1878             {
1879                 num.Add(value.MeasureId.Value);
1880             }
1881         }
1882     }
1883 }
```

```
1877     if (value.code_type == "N")
1878     {
1879         num.Add(value.MeasureId.Value);
1880     }
1881 }
1882
1883 submitted.numerators = num.ToArray();
1884 submitted.denominators = den.ToArray();
1885 submitted.exclusion = exc.ToArray();
1886 submitted.pending = pen.ToArray();
1887
1888 }
1889
1890 return Request.CreateResponse(HttpStatusCode.OK, submitted);
1891 }
1892 }
1893 catch (Exception ex)
1894 {
1895     return Request.CreateResponse(HttpStatusCode.InternalServerError, ex);
1896 }
1897
1898
1899 }
1900
1901 /// <summary>
1902 /// Below api is used to delete the gap document
1903 /// </summary>
1904 /// <param name="gapsid"></param>
1905 /// <param name="docid"></param>
1906 /// <param name="member"></param>
1907 /// <returns></returns>
1908 [HttpPost]
1909 [Route("api/ACO/deletegapdocument/{gapsid}/{docid}")]
1910 public HttpResponseMessage deletegapdocument(int gapsid, int docid, ACO_Member member)
1911 {
1912 }
```

```
ACOController.cs* UQP ProvidersController.cs Source Control Explorer ACO_MembersMart.cs BulkEmailController.cs Notification_UserSpecific.cs
UQP
1907 //<returns></returns>
1908 [HttpPost]
1909 [Route("api/ACO/deletegapdocument/{gapsid}/{docid}")]
1910 References
1911 public HttpResponseMessage deletegapdocument(int gapsid, int docid, ACO_Member member)
1912 {
1913     try
1914     {
1915         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
1916         {
1917             var query = db.ACODocsubmitteds.Where(e => e.ACOGapSubmittedId == gapsid && e.DocId == docid).FirstOrDefault();
1918             if (query != null)
1919             {
1920                 db.ACODocsubmitteds.Remove(query);
1921                 db.SaveChanges();
1922 
1923                 var docquery = (from g in db.AC_GapSubmitted
1924                               join d in db.ACODocsubmitteds
1925                               on g.gapsid equals d.ACOGapSubmittedId
1926                               where g.subscriberid == member.hicn && g.measureid == member.measureid
1927                               select new
1928                             {
1929                                 d.FileName,
1930                                 d.FilePath,
1931                                 d.DocId,
1932                                 d.ACOGapSubmittedId
1933                             }).ToList();
1934 
1935             List<ACO_Docs> doclist = new List<ACO_Docs>();
1936             foreach (var value in docquery)
1937             {
1938                 ACO_Docs docs = new ACO_Docs
1939                 {
1940                     filename = value.FileName,
1941                     filepath = value.FilePath,
1942                     documentid = value.DocId,
1943                     gapsid = value.ACOGapSubmittedId.Value
1944                 };
1945             }
1946         }
1947     }
1948 }
1949 }
```

```
1957     }
1958 }
1959
1960 /// <summary>
1961 // It will bring the listing which will be available to auditor
1962 // </summary>
1963 [HttpPost]
1964 [Route("api/ACO/auditlisting")]
1965 [References]
1966 public HttpResponseMessage auditlisting(ACO_Filter filter)
1967 {
1968     try
1969     {
1970         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
1971         {
1972             var measure = db.ACO_Measures.Select(e => new { e.Measure_ID, e.Measure_Name }).ToList();
1973             List<ACO_Member> memberlist = new List<ACO_Member>();
1974             ObjectParameter totalmembercount = new ObjectParameter("totalmembercount", typeof(Int32));
1975             int totalcount = 0;
1976             if(filter.documentstatus=="all")
1977             {
1978                 filter.documentstatus = null;
1979             }
1980             var query = db.ACO_Gaplisting(filter.enddate, filter.npi, filter.documentstatus, filter.subscribername, filter.hicn,
1981                 filter.measureid, filter.groupid, filter.assuranceid, filter.pageNumber, filter.pageSize, filter.report, filter.auditorid,
1982                 filter.dob, totalmembercount).ToList();
1983             totalcount = Convert.ToInt32(totalmembercount.Value);
1984             foreach (var value in query)
1985             {
1986                 ACO_Member member = new ACO_Member
1987                 {
1988                     subscriberfirstname = value.subscriberfirstname,
1989                     //subscribermidname = value.SubscriberMidName,
1990                     subscriberlastname = value.subscriberlastname,
1991                     dob = value.dob,
1992                     measureid = value.measureid,
1993                     measurename = (measure.Where(p => p.Measure_ID == value.measureid).Select(e => e.Measure_Name).FirstOrDefault() == null ? null :
```



```
controller.cs* ProvidersController.cs Source Control Explorer ACO_MemberMart.cs BulkEmailController.cs Notification_UserSpecific.cs
    ↴ * UQP.Controllers.ACOController
    ↴ auditing(ACO_Filter filter)

08 if (filter.report == false)
09 {
10     #region pagination
11     // Get's No of Rows Count
12     int count = totalcount;
13     //int count = lstmembermart.Count();
14
15     // Parameter is passed from Query string if it is null then it default Value will be pageNumber:1
16     int currentPage = filter.pageNumber;
17
18     // Parameter is passed from Query string if it is null then it default Value will be pageSize:20
19     int PageSize = filter.pageSize;
20
21     // Display TotalCount to Records to User
22     int TotalCount = count;
23
24     // Calculating Totalpage by Dividing (No of Records / Pagesize)
25     int TotalPages = (int)Math.Ceiling(count / (double)PageSize);
26
27     // Returns List of Customer after applying Paging
28     //var items = lstmembermart//.Skip((currentPage - 1) * PageSize).Take(PageSize).ToList(); []
29
30     // if currentPage is greater than 1 means it has previousPage
31     // var previousPage = currentPage > 1 ? "Yes" : "No";
32
33     // if TotalPages is greater than currentPage means it has nextPage
34     // var nextPage = currentPage < TotalPages ? "Yes" : "No";
35
36     // Object which we are going to send in header
37     var paginationMetadata = new
38     {
39         totalCount = TotalCount,
40         pageSize = PageSize,
41         currentPage = currentPage,
42         totalPages = TotalPages,
43         nextPage: "
```

```
UQP.Controllers.ACOController
2040     pageSize = pageSize,
2041     currentPage = currentPage,
2042     totalPages = totalPages,
2043     previousPage,
2044     nextPage
2045     };
2046
2047     // Setting Header
2048
2049     HttpContext.Current.Response.Headers.Add("Paging-Headers", Newtonsoft.Json.JsonConvert.SerializeObject(paginationMetadata));
2050     //HttpContext.Current.Response.Headers.Add("Filename", zipfileName);
2051     HttpContext.Current.Response.Headers.Add("Access-Control-Expose-Headers", "Paging-Headers");
2052     return Request.CreateResponse(HttpStatusCode.OK, memberlist);
2053 #endregion
2054 }
2055 else
2056 {
2057     var rpoj = new ReportsController()
2058     {
2059         Request = new System.Net.Http.HttpRequestMessage(),
2060         Configuration = new HttpConfiguration()
2061     };
2062     return rpoj.GenerateNotifReport(null, null, null, null, null, null, null, "ACO Audit", "excel", null, null, null, null, null,
2063         null, null, null, null, null, null, null, memberlist, null, null);
2064 }
2065
2066 }
2067
2068 }
2069 catch (Exception ex)
2070 {
2071     return Request.CreateResponse(HttpStatusCode.InternalServerError, ex);
2072 }
2073 }
```

```
2 //<summary>
3 /// Below api is used to accept or reject the aco gaps
4 /// </summary>
5 /// <param name="gaps"></param>
6 /// <returns></returns>
7 [HttpPost]
8 [Route("api/ACO/updategapstatus")]
9 References
0 public HttpResponseMessage Updategapstatus(ACO_Gaps gaps)
1 {
2     try
3     {
4         using (UQPEntities db = new UQPEntities(HttpContext.Current.Request.Headers.GetValues("IPA").FirstOrDefault()))
5         {
6             var query = db.AC0_Gapsubmitted.Where(e => e.gapsid == gaps.gapsid).FirstOrDefault();
7             if (query != null)
8             {
9                 var firstSpaceIndex = gaps.submittedfirstname.IndexOf(" ");
10                var firstname = gaps.submittedfirstname.Substring(0, firstSpaceIndex);
11                var lastname = gaps.submittedfirstname.Substring(firstSpaceIndex + 1);
12
13                query.documentstatus = gaps.documentstatus;
14                query.auditdate = DateTime.Today;
15                query.auditby = gaps.submittedby;
16                query.auditfirstname = firstname;
17                query.auditlastname = lastname;
18                db.SaveChanges();
19
20                ACOCComment comment = new ACOCComment
21                {
22                    userid = gaps.submittedby,
23                    gapid = gaps.gapsid,
24                    createdate = DateTime.Now,
25                    comment = gaps.comment,
26                };
27            }
28        }
29    }
30 }
```

Output from: Source Control - Team Foundation

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

- Open Files:** UQPController.cs, ProvidersController.cs, Source Control Explorer, ACO_MembersMart.cs, BulkEmailController.cs, and Notification_UserSpecific.cs.
- Current File:** UQPController.cs
- Code View:** The code is for the `ACOController` class, specifically the `PostGap` method. The code handles updating a gap record in the database and adding a comment to the `ACOComments` table.
- Code Snippet:**

```
2091 var lastname = gaps.submittedfirstname.Substring(firstSpaceIndex + 1);
2092
2093 query.documentstatus = gaps.documentstatus;
2094 query.auditdate = DateTime.Today;
2095 query.auditby = gaps.submittedby;
2096 query.auditfirstname = firstname;
2097 query.auditlastname = lastname;
2098 db.SaveChanges();
2099
2100 ACOComment comment = new ACOComment
2101 {
2102     userid = gaps.submittedby,
2103     gapid = gaps.gapsid,
2104     createdate = Datetime.Now,
2105     comment = gaps.comment,
2106     username = lastname + " " + firstname,
2107     Subscriberid = gaps.hicn,
2108     Measureid = gaps.measureid,
2109     Documentstatus = gaps.documentstatus,
2110     rolename = gaps.rolename
2111 };
2112 db.ACOComments.Add(comment);
2113 db.SaveChanges();
2114
2115 return Request.CreateResponse(HttpStatusCode.OK, "Gap updated successfully");
2116
2117 }
2118 catch (Exception ex)
2119 {
2120     return Request.CreateResponse(HttpStatusCode.InternalServerError, ex);
2121 }
2122 }
```
- Toolbars:** Standard, Debug, Tools, View, Project, and Solution.
- Status Bar:** Shows "100 %".
- Output Tab:** Shows "Show output from: Source Control - Team Foundation".