```
1   //Author: Dillon Wallace
2   //This file holds the logic for the Enhancement Calculations
3   //Contains:
4   //  1.) Formula for calcuating day (also one for month incase SEC decides to spread out
    enhancement over month to get more accurate payouts)
5   //  2.) todate turns into MONTH/YEAR string
6   //  3.) Payment logic to determine payrate based upon staffid
7   //  4.) PayOut Function that returns an object for the period and other information
8   //  5.) Prepare3MonthReport (maybe change) that prepares an array of data from
    crossfilter to report/upload Enhancement Data
9   //  6.) GenerateTable3Month creates tables for every record in the array produced by
    Prepare3MonthReport to display in a nice way.
10  //Works in progress:
11  //  7.) UploadEnhancementArray - Purpose: Uploads to mySQL Server data (single JSON
    File) so PHP can process and update
12  //          records in enhancemnet_Detail_temp so counts can be created
13  //  8.) Holds the function for Enhanacement Evts if elig
14
15  //Checks for service Elig.
16  function EnhancementEligService(d) {
17              if (d.SER.substring(0,1) >= "A" && d.SER.substring(0,1) <= "Z") {
18                 //console.log(d);
19                  return 1
20                  //console.log(d);
21              } else {
22                    SER = parseInt(d.SER);
23
24
25           if (((SER > 400 && SER <= 1999) || (SER >= 3000 && SER <= 999999) || SER ==
             240 || SER == 241 || SER == 229 || SER == 193 || SER == 773 || SER == 774 ||
             SER == 160 || SER == 232 || SER == 150 || SER == 228) && SER != 100 && SER !=
             420  && SER != 690) {
26           //console.log(d);
27           return 1
28
29          } else {
30           //console.log(d);
31           return 0
32
33          }
34          }
35  }
36
37
38
39
40  //Function to return everyday (from crossfilter) after 300 (5 hours) for enhancemnet
    payment processing in units
41  function ClinicalTimeIncentive (d, i) {
42  // d is object (array, key value pair)
43  // i is object/array index
44  var units;
45
46  //get total mins for day
47  var temp = d.value;
48  //total mins remaning after 360 (6 hours is removed)
49  /**
50  360 * totalWorkingDays. To compenstate across the month.
51  22 working days = 7920 mins
52  360 (6 hours)
53  300 (5 hours)
54
55  */
56  var remaining = temp - 300;
57
58  // if mins is not negative (greater than 6 hours that day do calucation else 0 units)
59  if (remaining > 0) {
60          // convert mins to units, if atleast one whole unit return whole number (no
            rounding)
```

```
61          if ((remaining/15) >= 1) {
62                  remaining = Math.floor(remaining/15);
63
64          } else {
65              remaining = 0;
66          }
67
68
69  } else {
70      remaining = 0;
71  }
72
73  units = remaining;
74
75  //turn to units in array for calulation
76  d.value = units;
77
78
79
80  }
81
82  //Function to return aggregated across month (from crossfilter) after 300 (5 hours) per
    day in month for enhancemnet payment processing in units
83  function ClinicalTimeIncentiveMONTH (d, i, hrs) {
84  // d is object (array, key value pair)
85  // i is object/array index
86  //minsToSubtract = hrs * 60;
87  var units;
88
89  //get total mins for day
90  var temp = d.value;
91  //total mins remaning after 360 (6 hours is removed)
92  /**
93  360 * totalWorkingDays. To compenstate across the month.
94  22 working days = 7920 mins
95  360 (6 hours)
96  300 (5 hours)
97
98  */
99  var remaining = temp - 300;
100
101 // if mins is not negative (greater than 6 hours that day do calucation else 0 units)
102 if (remaining > 0) {
103         // convert mins to units, if atleast one whole unit return whole number (no
            rounding)
104         if ((remaining/15) >= 1) {
105                 remaining = Math.floor(remaining/15);
106
107         } else {
108             remaining = 0;
109         }
110
111
112 } else {
113     remaining = 0;
114 }
115
116 units = remaining;
117
118 //turn to units in array for calulation
119 d.value = units;
120
121
122
123 }
124
125
126 //@date is a date type variable for calcuation to string
127 function toDateString(date) {
```

```
  var monthNames = [
  "January", "February", "March",
  "April", "May", "June", "July",
  "August", "September", "October",
  "November", "December"
];

//var date = new Date();
var day = date.getDate();
var monthIndex = date.getMonth();
var year = date.getFullYear();

return  monthNames[monthIndex] + " - " + year;
}




//@selectValue is the StaffID in a integer format for if logic
function CalcPay(selectValue) {
            if (selectValue >= 1000 && selectValue <= 1999) {
                    PayRate = 28.50;
            } else if (selectValue >= 4500 && selectValue <= 4999) {
                    PayRate = 17.00;
            } else {
                    PayRate = 13.50;
            }


    return PayRate;
}

//@grp is the grouped Dimention of DOS by day/month for calculation
function AggregateTime(grp) {
    //Returns the array for sum of current selection's mins by day.
    // Which will be used to get the units.
    rtrn = grp.reduceSum(function(d) {return d.duration});
    return rtrn;
}

//@arr is the array of grouped mins per day for clinicalTimeCalculations
function ClinicalTimeUnitCalc(arr) {
    Units = 0;
    //takes grouped array from crossfilter then runs calclations to determin units
    arr.all().forEach(ClinicalTimeIncentive);
    arr.all().forEach(function(d,i) {
    //debugging
//  console.log("Units Now: " + Units + " Current array: " + i + " Value: " + d.value);
    Units = Units + d.value;

});
return Units;
}


function PayOut(ndx, DayGrp, span) {
    var instance = {};
    var title, group;
    var dataGrp;
//  var DayGrp, Day;
//  var Units, PayRate, TotalPay;
    var DollarFormatter = d3.format(",.2f")
    //var _ndx;
    //      _ndx = ndx;
        //  group = ndxGroup;
    instance.filterAll = function() {
                this.redraw();
    }
```

```
197
198         instance.redraw = function() {
199                 Units = 0;
200                 PayRate = 0;
201                 TotalPay = 0;
202
203                 selectValue = d3.select('select').property('value');
204                 PayRate = CalcPay(selectValue);
205
206                 Day = AggregateTime(DayGrp);
207                 Units = ClinicalTimeUnitCalc(Day);
208
209         //      console.log("Staff ID: " + selectValue + " Mins: " + Units);
210                 TotalPay = Units * PayRate;
211
212             document.getElementById("Units").textContent = Units;
213             document.getElementById("Rate").textContent = PayRate;
214             document.getElementById("Paid").textContent = DollarFormatter(TotalPay);
215
216
217         /*  console.log("redrawn");
218             console.log(Day.all());
219             console.log(DayGrp.top(Infinity));
220             console.log("Units " + Units + " PayRate " + PayRate + " TotalPay " + TotalPay);
221             //console.log(group.reduceCount().value()); */
222
223
224             return instance;
225         }
226
227         instance.render = function() {
228                 Units = 0;
229                 PayRate = 0;
230                 TotalPay = 0;
231                 /*
232
233
234
235
236                 Day = DayGrp.reduceSum(function(d) { return d.duration });
237                 Day.all().forEach(ClinicalTimeIncentive);
238
239                 Day.all().forEach(function(d,i) { Units = Units + d.value });
240                 TotalPay = Units * PayRate;
241             */
242
243
244
245
246
247
248
249
250
251             return instance;
252         }
253         return instance;
254
255     }
256
257
258     //@StaffID is the array of staffID's.
259     //@staffIDim is the staffID Dimension for crossfilter to filter
260     //@DayDim is the dimension for DOS to group and reduce mins
261     //Will create 8 new rows in array object (U1,U2,U3 with units of month) (t1,t2,t3 holds
        month descirption) (total and rate)
262     //
263     function prepare3MonthReport(StaffID, staffIDDim, DayDim) {
264         //for each staff member;
```

```
StaffID.forEach(function(staff,staffI) {
    U1 = 0;
  U2 = 0;
  U3 = 0;
  var t1, t2, t3;
//create only one taff for calc
staffIDDim.filter(staff.STAFFID);
//create group
DayGrp = DayDim.group(d3.time.day).reduceSum(function(d) { return d.duration });



grp = DayGrp.all();

grp.forEach(ClinicalTimeIncentive);

grp.forEach(function(d,i) { d.month = d.key.getMonth()+1; });

var U1, U2, U3;
var t1, t2, t3;

grp.forEach(function(d,i) {

  if (i == 0) {
    // first iteration
    Counter = 1;
    tmpMonth = d.month;
    tmpDay = d;
    Units = 0;
    U1 = 0;
    U2 = 0;
    U3 = 0;
 //   var t1,t2,t3;
    t1 = "";
    t2 = "";
    t3 = "";
  }

  if (tmpMonth != d.month) {



        if (Counter == 2) {
      U2 = Units;
          t2 = toDateString(tmpDay.key);
          tmpDay = d;
       Counter = Counter + 1;
    }

    if (Counter == 1){
      t1 = toDateString(tmpDay.key);
      U1 = Units;
      tmpDay = d;
       Counter = Counter + 1;
    }




    //Reset Units, Month, and add Counter
    Units = 0;

        tmpMonth = d.month;


  }

```

```
    Units = Units + d.value;

            if (grp.length - 1 == i) {
              t3 = toDateString(d.key);
            U3 = Units;
      }




});

 var prate =  CalcPay(staff.STAFFID);

  staff.U1 = U1;
  staff.U2 = U2;
  staff.U3 = U3;
  staff.t1 = t1;
  staff.t2 = t2;
  staff.t3 = t3;
  staff.UTot = U1 + U2 + U3;
  staff.Rate = CalcPay(staff.STAFFID);

  staff.U1Pay = U1 * prate;
  staff.U2Pay = U2 * prate;
  staff.U3Pay = U3 * prate;

  staff.PTot = (U1 + U2 + U3) * prate;




});
}

//@StaffID is a preped array from prepare3MonthReport
function GenerateTable3Month(StaffID) {
    var DollarFormatter = d3.format(",.2f");
    StaffID.forEach(function(d,i) {
        content = '<table class="ui compact table">' +
'<thead> <tr><th colspan="4" style="text-align: left;">'+ d.Name + " - " + d.STAFFID +
'</th>'+
'</tr> <tr style="text-align: center;"> <th>Month-Year</th> <th>Units Earned</th>
<th>Rate</th> <th>Earned for Month</th> </tr> </thead>' +
'<tbody>' +
'<tr style="text-align: center;">'+
    '<td>' + d.t1 +'</td>'+
    '<td>' + d.U1 + '</td>' +
    '<td>$' + DollarFormatter(d.Rate) + '</td>' +
    '<td>$' + DollarFormatter(d.Rate * d.U1) + '</td>' +
'</tr>' +

'<tr style="text-align: center;">' +
    '<td>' + d.t2 +'</td>' +
    '<td>' + d.U2 + '</td>' +
    '<td>$' + DollarFormatter(d.Rate) + '</td>' +
    '<td>$' + DollarFormatter(d.Rate * d.U2) + '</td>' +
'</tr>' +

'<tr style="text-align: center;">' +
    '<td>' + d.t3 +'</td>' +
    '<td>' + d.U3 + '</td>' +
    '<td>$' + DollarFormatter(d.Rate) + '</td>' +
    '<td>$' + DollarFormatter(d.Rate * d.U3) + '</td>' +
'</tr>' +

'</tbody>' +
'<tfoot>' +
'<tr style="text-align: center;">' +
```

```
401       '<td><b>Totals:</b></td>' +
402       '<td>' + d.UTot + '</td>' +
403       '<td></td>' +
404       '<td>$' + DollarFormatter(d.UTot * d.Rate) + '</td>'+
405   '</tr> </tfoot> </table>'
406
407
408           $("#tableHolder").append(content);
409       });
410
411   }
412
413   //DONT USE JUST FOR LOOK
414   //Implement in file to control UI
415   /*function UploadEnhancementArray(StaffID) {
416
417                 var t = $.ajax({
418     url: "./PHP/Enhancement_Upload.php",
419     method: "POST",
420     data: JSON.stringify(StaffID), contentType: "application/json", processData: "false"
421   });
422
423   t.done(function() {
424
425
426   });
427
428   t.fail(function(jqXHR, textStatus) {
429   alert("Request failed: " + textStatus); });
430   }*/
431
432
```