```
<!doctype html>
<html>
<head>
<title id="page_title">fireplace_mkturk</title>
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script src="mkturk_installsettings.js"></script>
<script src="fireplace_googlecharts.js" type="text/javascript"></script>

<!-- ========== (begin) FIREBASE + FIRESTORE ========== -->
<!--   Firebase App is always required and must be first -->
<script src="https://www.gstatic.com/firebasejs/6.6.1/firebase.js"></script>
<script src="https://www.gstatic.com/firebasejs/6.6.1/firebase-app.js"></script>

<!-- Add additional services that you want to use -->
<script src="https://www.gstatic.com/firebasejs/6.6.1/firebase-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/6.6.1/firebase-firestore.js"></script>

<script src=https://cdnjs.cloudflare.com/ajax/libs/mathjs/3.3.0/math.min.js></script>
<script type="text/javascript">
  // Initialize Firebase
  var config = {
    apiKey: "AIzaSyA0fbv2VqE-AfF6V_nxSSXCEqaTlBlZnTI",
    authDomain: "sandbox-ce2c5.firebaseapp.com",
    databaseURL: "https://sandbox-ce2c5.firebaseio.com",
    projectId: "sandbox-ce2c5",
    storageBucket: "sandbox-ce2c5.appspot.com",
    messagingSenderId: "1003719887944"
  };
  firebase.initializeApp(config);

  //Initialize Cloud Firestore
  var db = firebase.firestore();
</script>
<!-- ========== (end) FIREBASE + FIRESTORE ========== -->


<!-- ========== (begin) MATERIAL DESIGN LITE ========== -->
<!-- Include Material Design Lite CDN hosted components -->
<!-- Getting started @ https://getmdl.io/started/index.html -->
<!-- MDL color theme picker @ https://getmdl.io/customize/index.html -->
    <script src="https://code.getmdl.io/1.3.0/material.min.js"></script>
    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Roboto:regular
,bold,italic,thin,light,bolditalic,black,medium&amp;lang=en">
    <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icon
s">
    <link rel="stylesheet" href="https://code.getmdl.io/1.3.0/material.deep_purple-pink
.min.css">
<!--     <link rel="stylesheet" href="styles.css"> -->
<!-- ========== (end) MATERIAL DESIGN LITE ========== -->

<script type="text/javascript">
// Flow:
// authenticate --> loadGoogleCharts --> queryFirestore -->

// // FIREPLACE modifications
//---- (3a) live updates - https://firebase.google.com/docs/firestore/query-data/listen
//---- (3b) expanded table (last trial, battery level, last ping, green if running toda
y)
// (4) plot other vars, pulldown: choice bias, target accuracy, weight, all params

// // EASY TO IMPLEMENT
// # of objects
// sample on time
// reward amount
// punish timeout
```

```
// sample & test scales
// battery used per hour

// // MORE INVOLVED
// choice bias
// total # of rewards
// touch accuracy

//================== GOOGLE CHARTS LOADING (begin) ==================//
// Asynchronous: Live plotting using google charts
function loadGoogleCharts() {
    // Load the Visualization API and the piechart package.
    google.charts.load('current', {'packages': ['corechart','line', 'bar', 'table', 'co
ntrols'] });
    google.charts.setOnLoadCallback(function() {
        dataPerformance = new google.visualization.DataTable()
        dataTrial = new google.visualization.DataTable()
        dataLeaderboard = new google.visualization.DataTable()

        liveline = new google.charts.Line(document.getElementById('line_div'));
        livelinetrial = new google.charts.Line(document.getElementById('linetrial_div')
);
        livetable = new google.visualization.Table(document.getElementById('table_div')
);

        google.visualization.events.addListener(liveline, 'select', selectHandlerPerfor
mance1);
        google.visualization.events.addListener(livelinetrial, 'select', selectHandlerP
erformance2);

        queryFirestore(null) //ndays back
    })
}

function selectHandlerPerformance1() {
  console.log('selectHanderPerformance1()');
  var selection = liveline.getSelection();
  var message = '';
  for (var i = 0; i < selection.length; i++) {
    var item = selection[i];
    if (item.row != null && item.column != null) {
      var str = dataPerformance.getFormattedValue(item.row, item.column);
      message += '{row:' + item.row + ',column:' + item.column + '} = ' + str + '\n';
      showTaskText(item.row,item.column)
    }
  }
  console.log('User selected ' + message);
}

function selectHandlerPerformance2() {
  console.log('selectHandlerPerformance2()');
  var selection = livelinetrial.getSelection();
  var message = '';
  for (var i = 0; i < selection.length; i++) {
    var item = selection[i];
    if (item.row != null && item.column != null) {
      var str = dataPerformance.getFormattedValue(item.row, item.column);
      message += '{row:' + item.row + ',column:' + item.column + '} = ' + str + '\n';
      showTaskText(item.row,item.column)
    }
  }
  console.log('User selected ' + message);
}

function showTaskText(dayidx,agentidx){
```

```
  console.log('showTaskText()');
  console.log('day ' + new Date(STAT.t[dayidx]).toDateString()  + ', Agent ' + STAT.age
nts[agentidx-1])

  //add object metadata
  var task = DATA.task[STAT.lastfile[agentidx-1][dayidx]];
  var imagesRef = db.collection(FIRESTORECOLLECTION.DATA).doc(task.Imagesdoc)

  imagesRef.get().then(
    function(doc)
    {
      var taskname = ''
      if (task.RewardStage == 0){
        taskname = 'FIXATION'
      }
      else if (task.TestON > 0){
        taskname = 'SD' + doc.data().TestNouns.length
      }
      else if (task.ObjectGridIndex.length == task.TestGridIndex.length)
      {
        taskname = 'SR' + task.ObjectGridIndex.length
      }
      else if (task.HideTestDistractors == 0){
        taskname = 'M2S' + doc.data().TestNouns.length + ' R' + task.TestGridIndex.leng
th
      }

      var textobj = document.getElementById("tasktext");
      textobj.innerHTML =
      "<b>" + taskname + "</b>"
      +"<br> <b>Sample nouns:</b> " + doc.data().SampleNouns + '  (' + doc.data().Sampl
eBagNames + ')'
      + "<br> <b>Test nouns:</b> " + doc.data().SampleNouns + '  (' + doc.data().TestBa
gNames + ')'
      + "<br>"
      + "<br> Sample object models: " + doc.data().SampleObjects
      + "<br> Test object models: " + doc.data().TestObjects
      + "<br><br>"
      + JSON.stringify(task, function(a, b) {
        return (Object.prototype.toString.call(b) === '[object Array]') ? JSON.stringif
y(b) : b;
      }, 4)
    })
}
//================== GOOGLE CHARTS LOADING (end) ==================//


//================== FIRESTORE QUERY (begin) ==================//
async function queryFirestoreCallback(querySnapshot){
  console.log('queryFirestoreCallback()');
//NOTE: would use generator to step out for X seconds (pause), but firestore externally
 calls query Firestore anyway, so only get new data & update plot every X seconds, but
can't prevent multiple calls from firestore in the meantime
  if (performance.now() - tlastquery < queryUpdateInterval && queryfirstpass == 0){
    console.log('NEW DATA! Skipping snapshot callback @' + Math.round(performance.now()
-tlastquery) + 'ms')
    return //wait queryUpdateInterval milliseconds between updates
  }
  queryfirstpass = 0
  console.log('NEW DATA! Loading... @' + Math.round(performance.now()-tlastquery) + 'ms
')

  document.querySelector('#progressbar').style.display = "block"
  document.querySelector('#progressbar').style.visibility = "visible"
  document.getElementById('titletext').innerHTML =
```

```
              'fireplace mkturk   ' + "<font size=-4><font color=#673ab7>" + 'U
pdating...' + "</font>"

  var ndocsquerytotal = querySnapshot.size;
  DATA.response = []
  DATA.correctitem = []
  DATA.ncorrect = []
  DATA.agent = []
  DATA.date = []
  DATA.t = []
  DATA.task = []
  DATA.ndocs = 0

  querySnapshot.forEach(function(doc)
  {
      if (typeof(doc.data().Response) == "undefined"){
          // do nothing
          if (doc.data().Doctype == "task"){
              console.log('Doc contains no response data, skipping')
          }
      }
      else if (doc.data().Agent == "Eliaso" || doc.data().Agent == "Sophieo"
                || doc.data().Agent == "Youno" || doc.data().Agent == "Hectoro"
                || doc.data().Agent == "Ericao" || doc.data().Agent == "Tahereho"
                || doc.data.Agent == "Alexo"){
        console.log('Doc is debug file of Eliaso/Sophieo/Youno, skipping')
      }
      else //contains trial data
      {
          DATA.agent[DATA.ndocs] = doc.data().Agent
          DATA.date[DATA.ndocs] = doc.data().CurrentDate

          // # of days from today
          // round to times to nearest day & add 8 hours
          DATA.t[DATA.ndocs] = (24*60*60*1000) * Math.floor((doc.data().CurrentDate.toD
ate().valueOf())/(24*60*60*1000)) + 0.25*(24*60*60*1000);

          var r = math.matrix(doc.data().Response);
          var c = math.matrix(doc.data().CorrectItem.splice(0,r.size()[0]));
          var ncorrect = math.filter(math.subtract(r,c), el => el==0).size()[0];

          DATA.response[DATA.ndocs] = doc.data().Response
          DATA.correctitem[DATA.ndocs] = doc.data().CorrectItem
          DATA.ncorrect[DATA.ndocs] = ncorrect

          DATA.task[DATA.ndocs] = doc.data()

          DATA.lastdatevalue[DATA.ndocs] = doc.data().CurrentDateValue + doc.data().Sta
rtTime[doc.data().StartTime.length-1]

          if (Array.isArray(doc.data().BatteryLDT[0]) == true){
              var nupdates = doc.data().BatteryLDT[0].length
              DATA.lastbatterypct[DATA.ndocs] = 100*doc.data().BatteryLDT[0][nupdates-1
]
              DATA.batterypctperhour[DATA.ndocs] = 100*1000*60*60*
                  (doc.data().BatteryLDT[0][nupdates-1] -
                  doc.data().BatteryLDT[0][0]) /
                  (doc.data().BatteryLDT[2][nupdates-1] - doc.data().BatteryLDT[2][0])
              DATA.batterypctperhour[DATA.ndocs] = 0.1*Math.round(10*DATA.batterypctper
hour[DATA.ndocs])
          }
          else {
              DATA.lastbatterypct[DATA.ndocs] = 0
              DATA.batterypctperhour[DATA.ndocs] = 0
          }
```

```
          DATA.ndocs++

          console.log(doc.id, " ==> ", doc.data().CorrectItem.length
           + ' trials, ' + Math.round(100*ncorrect/doc.data().Response.length) + "%");
      } //if contains data
  }); // forEach doc

    //Delete some of task metadata & trialdata to save space
    for (var j=0; j<=DATA.task.length-1; j++){
      delete DATA.task[j].CorrectItem
      delete DATA.task[j].Response
      delete DATA.task[j].FixationXYT
      delete DATA.task[j].ResponseXYT
      delete DATA.task[j].FixationGridIndex
      delete DATA.task[j].FixationTouchEvent
      delete DATA.task[j].ResponseTouchEvent
      delete DATA.task[j].NReward
      delete DATA.task[j].Sample
      delete DATA.task[j].Test
      delete DATA.task[j].StartTime
    }
    console.log(querySnapshot.size + " documents recovered")

    //store position of query
    DATA.queries.start.push(startdatevalue)
    DATA.queries.end.push(enddatevalue)

    joinData()
    toGoogleDataTable()

    //update axes range
    lineOptions.hAxis = {viewWindow: {
            min: new Date(startdatevalue),
            max: new Date(enddatevalue)
      }
    }
    linetrialOptions.hAxis = {viewWindow: {
            min: new Date(startdatevalue),
            max: new Date(enddatevalue)
      }
    }
    liveline.draw(dataPerformance,google.charts.Line.convertOptions(lineOptions))
    livelinetrial.draw(dataTrial,google.charts.Line.convertOptions(linetrialOptions))
    livetable.draw(dataLeaderboard,google.charts.Line.convertOptions(tableOptions))

    DATA.totalloadtime = performance.now() - tlastquery
    tlastquery = performance.now()

    var d = new Date()
    h = d.getHours()
    m = d.getMinutes()
    s = d.getSeconds()
    var ampm = 'am'
    if (h>12){ h = h-12; ampm = 'pm' }
    if (m<10){ m = "0" + m}
    if (s<10){ s = "0" + s}
    document.getElementById('titletext').innerHTML =
            'fireplace mkturk   ' +
            "<font size=-4>" + DATA.ndocs + ' docs, ' + Math.round(DATA.totalloadtime/1
000) + ' sec' + "</font>" +
            "<font size=-4><i>" + '   ' + h + ':' + m + ':' + s + '' + ampm + "
</i></font>"
    document.querySelector('#progressbar').style.display = "none"
} // FUNCTION queryFirestoreCallback(querySnapshot)
```

```
async function queryFirestore(event){
  console.log('queryFirestore()');
  if (event != null){
    event.preventDefault();
  }
  if (typeof(query) != "undefined"){
    query() //unsubscribe from last query
  } //if query active

  queryfirstpass = 1
  document.querySelector('#progressbar').style.display = "block"
  document.querySelector('#progressbar').style.visibility = "visible"
  document.getElementById('titletext').innerHTML =
              'fireplace mkturk   ' + "<font size=-4><font color=#673ab7>" + 'U
pdating...' + "</font>"
  query = db.collection(FIRESTORECOLLECTION.DATA)
            .where("Doctype","==",'task')
            .where("CurrentDateValue",">=",startdatevalue)
            .where("CurrentDateValue","<=",enddatevalue).onSnapshot(querySnapshot => qu
eryFirestoreCallback(querySnapshot))
} //FUNCTION initialize firestore query, kick off listener
//================== (end) FIRESTORE QUERY ==================//


//================== (begin) DATA JOIN ==================//
function joinData(){
  console.time('datajoin')
    // determine # of days and # of agents
    STAT.agents = [...new Set(DATA.agent)] //unique agents
    STAT.agents.sort() //alphabetical order
    STAT.tlasttrial = Array(STAT.agents.length).fill(0)
    STAT.lastbatterypct = Array(STAT.agents.length).fill(0)
    STAT.batterypctperhour = Array(STAT.agents.length).fill(0)

    STAT.t = [...new Set(DATA.t)] //unique days
    STAT.t.sort()

    STAT.ntrials = Array(STAT.agents.length).fill([]).map(element => {return Array(STAT
.t.length).fill(0)})
    STAT.pct = Array(STAT.agents.length).fill([]).map(element => {return Array(STAT.t.l
ength).fill(0)})
    STAT.firstfile = Array(STAT.agents.length).fill([]).map(element => {return Array(ST
AT.t.length).fill(-1)})
    STAT.lastfile = Array(STAT.agents.length).fill([]).map(element => {return Array(STA
T.t.length).fill(-1)})
    for (var i=0; i<=DATA.response.length-1; i++){
        var agentidx = STAT.agents.indexOf(DATA.agent[i])
        var dayidx = STAT.t.indexOf(DATA.t[i])

        STAT.pct[agentidx][dayidx] =
            ((0.01*STAT.pct[agentidx][dayidx]*STAT.ntrials[agentidx][dayidx]) + DATA.nc
orrect[i])
        STAT.ntrials[agentidx][dayidx]+= DATA.response[i].length
        STAT.pct[agentidx][dayidx] = 100*STAT.pct[agentidx][dayidx]/STAT.ntrials[agenti
dx][dayidx]

        if (STAT.firstfile[agentidx][dayidx] == -1){
            STAT.firstfile[agentidx][dayidx] = i
        }
        STAT.lastfile[agentidx][dayidx] = i

        if (STAT.tlasttrial[agentidx] < DATA.lastdatevalue[i]){
          STAT.tlasttrial[agentidx] = DATA.lastdatevalue[i]
          STAT.lastbatterypct[agentidx] = DATA.lastbatterypct[i]
          STAT.batterypctperhour[agentidx] = DATA.batterypctperhour[i]
```

```
        } //if more recent file for that agent
    } //for i docs
console.timeEnd('datajoin')
}
//================== (end) DATA JOIN ==================//

// ================== (begin) DATA HANDLING ==================//
// Synchronous
function toGoogleDataTable() {
    //-------- Performance line chart ---------//
    dataPerformance.removeColumns(0,dataPerformance.getNumberOfColumns());
    dataPerformance.removeRows(0,dataPerformance.getNumberOfRows());

    dataPerformance.addColumn('datetime','')
    for (var i=0; i<=STAT.agents.length-1; i++){
        dataPerformance.addColumn('number',STAT.agents[i]);
    }

    //Create the data table
    for (var i = 0; i <= STAT.pct[0].length - 1; i++){
        var array_t = []
        for (var j =0; j<= STAT.pct.length-1; j++){
            array_t[j] = STAT.pct[j][i]
        } //for j agents
        dataPerformance.addRows([[new Date(STAT.t[i]),...array_t]])
    } //for i timepoints

    //-------- Trial line chart ---------//
    dataTrial.removeColumns(0,dataTrial.getNumberOfColumns());
    dataTrial.removeRows(0,dataTrial.getNumberOfRows());

    dataTrial.addColumn('datetime','')
    for (var i=0; i<=STAT.agents.length-1; i++){
        dataTrial.addColumn('number',STAT.agents[i]);
    }

    //Create the data table
    for (var i = 0; i <= STAT.ntrials[0].length - 1; i++){
        var array_t = []
        for (var j =0; j<= STAT.ntrials.length-1; j++){
            array_t[j] = STAT.ntrials[j][i]
        } //for j agents
        dataTrial.addRows([[new Date(STAT.t[i]),...array_t]])
    } //for i timepoints


    //-------- Leaderboard table ----------//
    var ndays = STAT.pct[0].length
    dataLeaderboard.removeColumns(0,dataLeaderboard.getNumberOfColumns());
    dataLeaderboard.removeRows(0,dataLeaderboard.getNumberOfRows());
    dataLeaderboard.addColumn('string','Agent');
    dataLeaderboard.addColumn('number','% (today)');
    dataLeaderboard.addColumn('number','n (today)');
    dataLeaderboard.addColumn('number','% (-2d)');
    dataLeaderboard.addColumn('number','n (-2d)');
    dataLeaderboard.addColumn('number','tlast (min)');
    dataLeaderboard.addColumn('number','batt%');
    dataLeaderboard.addColumn('number','%/hr');

    for (var i = 0; i <= STAT.agents.length-1; i++){
      var ntotal = 0
      var ncorrect = 0
      var nsess = 0

      // Get performance for previous two sessions
```

```
      for (var j = 1; j <= STAT.pct[i].length-1; j++){
        if (nsess >= 2){
          break;
        }
        var ind = STAT.pct[i].length - j - 1
        if (STAT.ntrials[i][ind] > 0){
          ntotal += STAT.ntrials[i][ind]
          ncorrect += STAT.ntrials[i][ind] * STAT.pct[i][ind]/100
          nsess+=1
        } //if
      } //for j days

      var tsincelasttrial = Math.round( ( new Date().valueOf() - STAT.tlasttrial[i] ) /
 1000 / 60 )

      dataLeaderboard.addRows([
        [ STAT.agents[i],
        Math.round(STAT.pct[i][ndays-1]), STAT.ntrials[i][ndays-1],
        Math.round(100*ncorrect/ntotal), Math.round(ntotal/nsess),
        tsincelasttrial,
        STAT.lastbatterypct[i], STAT.batterypctperhour[i],
        ]
      ])

      // Color based on last trial time
      tlastFormatter = new google.visualization.ColorFormat();
      tlastFormatter.addRange(0, 5, 'black', '#33ff33');
      tlastFormatter.addRange(5, 60, 'black', 'red');
      tlastFormatter.addRange(60, null, 'black', 'white');
      tlastFormatter.format(dataLeaderboard, 5); // Apply formatter to fourth column
      if (tsincelasttrial < 5){ //actively working
          dataLeaderboard.setCell(i, 0, STAT.agents[i],STAT.agents[i], {'style': 'color
: #33ff33; background-color: white; font-weight: bold'});
      }
      else if (tsincelasttrial < 60){ //was working but stopped recently
          dataLeaderboard.setCell(i, 0, STAT.agents[i],STAT.agents[i], {'style': 'color
: red; background-color: white; font-weight: bold'});
      }
      else{ //inactive
          dataLeaderboard.setCell(i, 0, STAT.agents[i],STAT.agents[i], {'style': 'color
: black; background-color: white'});
      }

      // Color based on battery life
      if (STAT.lastbatterypct[i] < 25){ //battery low
          dataLeaderboard.setCell(i, 6, STAT.lastbatterypct[i],STAT.lastbatterypct[i],
{'style': 'color: red; background-color: white; font-weight: bold'});
      }
    } //for i agents
}
//================== DATA HANDLING (end) ==================//


function queryduration_listener(event){
  queryndaysback = 7 * event.target.value; //convert weeks -> days
  console.log('query duration changed = ' + queryndaysback + ' days')
  startdatevalue = enddatevalue - queryndaysback*(24*60*60*1000)
  updateDateChip()
}

function queryend_listener(event){
  enddatevalue = new Date(event.target.value).valueOf()
  updateDateChip()
}
```

```
function updateDateChip(){
  var str = '<b>' + new Date(startdatevalue).toDateString() + ' - ' + new Date(enddatev
alue).toDateString() + '</b>'
  document.getElementById("datechip").innerHTML = str;
}

</script>
</head>

<body class="mdl-demo mdl-color--grey-100 mdl-color-text--grey-700 mdl-base">

<div>
<h4 id="titletext">fireplace mkturk</h4>
</div>

<!-- MDL Progress Bar with Indeterminate Progress -->
<div id="progressbar" class="mdl-progress mdl-js-progress mdl-progress__indeterminate">
</div>

<span class="mdl-chip">
    <span class="mdl-chip__text" id="datechip"> <b> DateRange </b> </span>
</span>

<!-- Numeric Textfield -->
<form action="#">
<div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
  <input class="mdl-textfield__input" type="text" pattern="-?[0-9]*(\.[0-9]+)?" id="que
rydurationinput">
  <label class="mdl-textfield__label" for="querydurationinput">Number of weeks back</la
bel>
  <span class="mdl-textfield__error">NaN!</span>
</div>
</action>

<!-- Numeric Textfield -->
<form action="#">
<div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
  <input class="mdl-textfield__input" type="text" id="queryendinput">
  <label class="mdl-textfield__label" for="queryendinput">End date</label>
  <span class="mdl-textfield__error">Input is not a number!</span>
</div>
</action>

<!-- Colored FAB button with ripple -->
<button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect mdl-button
--colored" id="runquerybutton">
  <i class="material-icons">refresh</i>
</button>

<div id="table_div"></div> <!--Div that will hold the leaderboard table-->
<p></p>
<div id="line_div"></div> <!--Div that will hold the line chart-->
<div id="linetrial_div"></div> <!--Div that will hold the line chart-->
<pre id="tasktext" style="position: absolute; left: 1px; width: 100%; font-size: 18px;
color: black; font-family: 'Helvetica Neue', Helvetica, Arial, sans-serif; word-break:
break-all;"></pre>
<script type="text/javascript">

document.getElementById("querydurationinput").addEventListener("change",queryduration_l
istener,false);
document.getElementById("queryendinput").addEventListener("change",queryend_listener,fa
lse);
document.getElementById("runquerybutton").addEventListener("pointerup",queryFirestore,f
alse)
```

```
var liveline = null;
var dataPerformance = null;
var formatterDate = null;
var formatterDigits = null;
var queryUpdateInterval = 20000 //milliseconds
var tlastquery = performance.now() - queryUpdateInterval

var DATA = {
    response: [],
    correctitem: [],
    ncorrect: [],
    agent: [],
    date: [],
    t: [],
    lastdatevalue: [],
    lastbatterypct: [],
    batterypctperhour: [],
    task: [],
    queries: {start: [], end: []},
    totalloadtime: 0,
    ndocs: 0,
};

var STAT = {
    agents: [],
    t: [],
    ntrials: [],
    pct: [],
    firstfile: [],
    lastfile: [],
    tlasttrial: [],
}

var GOOGLEUSER = {
    ResearcherDisplayName: "",
    ResearcherEmail: "",
    ResearcherLastName: "",
    ResearcherID: "",
};

var currdate = new Date()
var refdate = new Date(Date.UTC(currdate.getFullYear(),currdate.getMonth(),currdate.get
Date(),28,59,0)) //end of today in UTC which is +5 hours
var queryndaysback = 7
var enddatevalue = refdate.valueOf()
var startdatevalue = refdate.valueOf() - queryndaysback*(24*60*60*1000) //msperday
document.getElementById("querydurationinput").value = Math.round(queryndaysback/7)
document.getElementById("queryendinput").value = refdate.toLocaleDateString()
updateDateChip()


//================== (begin) AUTHENTICATE GOOGLEUSER ==================//
// self-executing anonymous function
// (async function(){
//     await firebaseToggleSignIn()
// })();

//================== AUTHENTICATE GOOGLE ==================//
// [START authstatelistener]
var provider = new firebase.auth.GoogleAuthProvider();
provider.addScope('https://www.googleapis.com/auth/contacts.readonly');
// firebase.auth().onAuthStateChanged(function(user) {
//   if (user) {
//         GOOGLEUSER.ResearcherDisplayName = user.displayName;
//         GOOGLEUSER.ResearcherEmail = user.email;
```

```
//          GOOGLEUSER.ResearcherID = user.uid
//          console.log('USER ' + user.email + ' signed in')
//    } else {
//      firebase.auth().signInWithRedirect(provider)
//    }
// });
firebase.auth().getRedirectResult().then(function(result) {
  if (result.user) {
    // User just signed in. you can get the result.credential.
    GOOGLEUSER.ResearcherDisplayName = result.user.displayName;
        GOOGLEUSER.ResearcherEmail = result.user.email;
        GOOGLEUSER.ResearcherID = result.user.uid

        console.log('Sign-In Redirect Result, USER ' + result.user.email + ' is signed
in')
  }
  else if (firebase.auth().currentUser) {
    // User already signed in.
        GOOGLEUSER.ResearcherDisplayName = firebase.auth().currentUser.displayName;
        GOOGLEUSER.ResearcherEmail = firebase.auth().currentUser.email;
        GOOGLEUSER.ResearcherID = firebase.auth().currentUser.uid

        console.log('Already Signed In, USER ' + firebase.auth().currentUser.email + '
is signed in')
  }
  else {
    // No user signed in, update your UI, show the redirect sign-in screen.
        firebase.auth().signInWithRedirect(provider)
  }
});
//================== (end) AUTHENTICATE GOOGLE ==================//


loadGoogleCharts();

</script>
</body>
</html>
```