// Hide form to disallow checkin

$("form").hide();

// Change message to show we're tracking their location

$("p#message").html("Tracking your location now... :)");

// Find the users location using geolocation

watchUser = navigator.geolocation.watchPosition(success, error);

// Success is run when watchPosition is successful

function success(position){

// Test if tracking worked in browser console

console.log("Tracking was successful!");

// View position object in browser console

console.log(position);

// Capture user location coordinates in variables

var userLat = position.coords.latitude;

var userLon = position.coords.longitude;

//store the target location variables

var targetLat = 51.586643;

var targetLon = -0.149774;

// calculate the distance

var distance = getDistanceFromLatLonInKm(userLat,userLon,targetLat,targetLon);

// check distance is sensible;

console.log("distance = " + distance + "km")

// set radius

var radius = 0.5; // 0.5km = 500m

if (distance<radius) {

// stuff to do if true

// Show form to allow checkin

$("form").show();

// Change message to say hello

$("p#message").html("Hello! Please check in with your twitter handle above :)");

} else {

// stuff to do if NOT true (ie false)

// Hide form to disallow checkin

$("form").hide();

// Change message to encourage user

$("p#message").html("We're waiting for you, your distance from Highgate Woods is " + distance.toFixed(2) + "km");

}

} // END success

function error() {

alert("Sorry, I can't find you! Please enable geolocation")

} // END error function

// code from stack overflow

function getDistanceFromLatLonInKm(lat1,lon1,lat2,lon2) {

var R = 6371; // Radius of the earth in km

var dLat = deg2rad(lat2-lat1); // deg2rad below

var dLon = deg2rad(lon2-lon1);

var a =

Math.sin(dLat/2) \* Math.sin(dLat/2) +

Math.cos(deg2rad(lat1)) \* Math.cos(deg2rad(lat2)) \*

Math.sin(dLon/2) \* Math.sin(dLon/2)

;

var c = 2 \* Math.atan2(Math.sqrt(a), Math.sqrt(1-a));

var d = R \* c; // Distance in km

return d;

}

function deg2rad(deg) {

return deg \* (Math.PI/180)

}