Delivery plan sprint-1

Live Location Tracking:

GPS is installed on gadget to track its current location can be tracked on android app and via SMS request sent from parent phone to safety gadget. Outputs of live location tracking

2) Panic Alert Systems:

Panic alert system on gadget is triggered during panic situation, automatic call and SMS are triggered to parental phone. The alert is also updated to the cloud for purpose of app monitoring. Fig. 4. Outputs of panic alert system.

3) Stay Connected Feature:

Stay connected feature is used to trigger call and pre-defined SMS anytime from gadget to parental phone by just pressing a button and also parent can make SMS and call to the gadget anytime.

4) Health Monitoring System:

Health monitoring system is implemented using heart beat sensor, temperature sensor which is updated to the cloud and also can be monitored via app. The current value of sensors can be obtained using SMS request sent to gadget from parent phone. Outputs of health monitoring system.

5) Gadget Plugged or Unplugged Monitoring:

Gadget plug or unplugged is monitored using contact switch installed on smart gadget, as soon as the device is unplugged, an alert is provided to parent phone via SMS and it is also updated to cloud for app monitoring.

GEOFENCING CODE:

Basic Example Code:

```
 import time
def
        stopwatch(secon
        ds,d,lspoint): start
        = time.time()
        time.clock()
         elapsed
        =  0 flag
        =   False
        num = 0
         while elapsed < seconds:
                 elapsed              =
                time.time()  -  start
                print    "%02d"    %
                elapsed

                if elapsed > d[num] and elapsed < d[num+1]
                        and flag == False: x = lspoint[num][0]

                         y =
                        lspoint[num]
                        [1]
                        createpoint(x
            ,y) flag =
            True
             print "Shot Taken"
            print
        point_in_poly(x,y,polygon)
        if elapsed > d[num+1]:
             print   "Shot
            Taken"   flag
            == False
             num = num+1
             x              =
            lspoint[num]
            [0]     y      =
            lspoint[num]
            [1]
            createpoint(x
            ,y)
            print
        point_in_poly(x,y,polygon)
        time.sleep(1)

def createpoint(x,y):
   crs =
  "point?crs=epsg:27700&field=id:i
  nteger" layer =
  QgsVectorLayer(crs, 'points' ,
  "memory")pr =
  layer.dataProvider()
   pt =
  QgsFeature()
  point1 =
  QgsPoint(x,y)

   pt.setGeometry(QgsGeometry.fromP
  oint(point1))pr.addFeatures([pt])

   # update extent of
  the layer
  layer.updateExtent
  s()

   # add the
  second point pt
  = QgsFeature()

   QgsMapLayerRegistry.instance().addMapLayers([layer])


def point_in_poly(x,y,poly):
 n =
len(poly)
inside =
False

 p1x,p1y =
poly[0] for i
in
range(n+1):
    p2x,p2y =
   poly[i % n]if y
   >
   min(p1y,p2y):
       if y <= max(p1y,p2y):
         if x <=
            max(p1x,p2x
            ):if p1y !=
            p2y:
               xints = (y-p1y)*(p2x-
            p1x)/(p2y-p1y)+p1x if p1x ==
            p2x or x <= xints:
               inside = not inside p1x,p1y =
            p2x,p2y


    return inside

#### define the polygon
 polygon =
[(512882.78819722467,120811.83924772343),(512960.8443717052
6,120809.7007223952),(512960.
 84437170526,120809.7007223952),(512959.77510904113,120754.0
9906386107),(512882.78819722
 467,120756.2375891893)]


 #### set how long the script will run (70 seconds will get

you in and out of geofence) time_seconds = 70

 #### first

coordinate x =

512915

 y = 120728
 #### time intervals, 10 seconds

between shots / or points intervals =

int(time_seconds / 10)

 lspoint = []
 #### build the list of

coordinates to be plotted for i in

range(0,intervals+1):

        y1 = y +
        (i*12.5)

        lspoint.append(

        [x,y1])


 #### to build the blocks of time in intervals, so we know the number
of intervals (default is 7),

 #### we need a list of time intervals [0,10,20,30 etc] to check
against the clock this list is d, f is the gap ie 10 seconds, a is starting
point (0)

 ### b is the number of intervals + 1 becuase the code will

check the the next in the list f = 10

 a = 0

 b = intervals+1

 d = [x * f for x in range(a, b)]


 ### Run the stopwatch, or

start the program!

stopwatch(time_seconds,d,lsp

oint)
```