Aman Ladia's

Maker Portfolio

Living in Mumbai for 12 years, I have witnessed my city's unsustainable growth. This has compelled me to use my prototyping & design skills to develop products that bring sustainability and security to growing settlements. I have tackled not only the physical aspects of this theme, but also the virtual ones (privacy) through my projects below:

ZeroWallet

ILLINOIS
Coordinated Science Laboratory
GRAINGER COLLEGE OF ENGINEERING

June 2019 Onward
Developed with
Dr Andrew Miller, UIUC

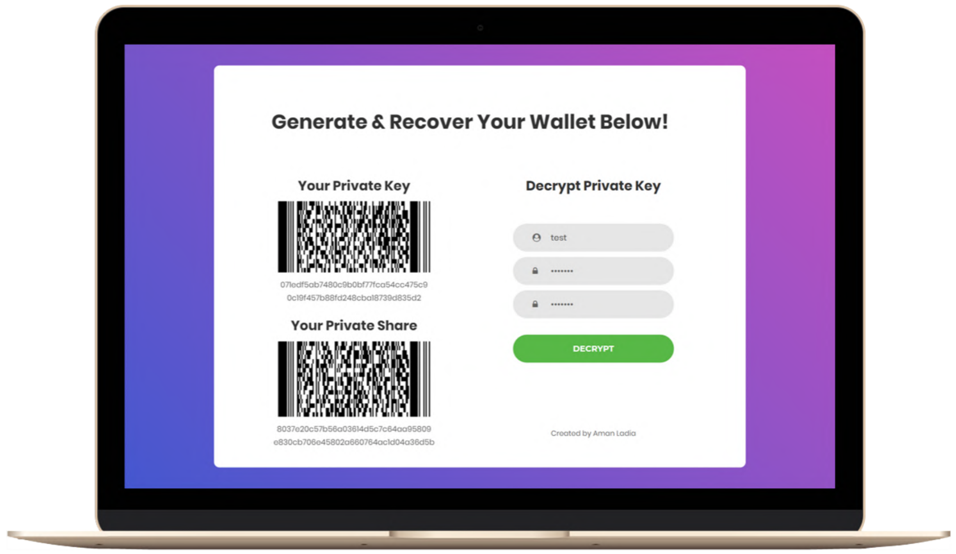
Zerowallet.me

Merging the convenience of PayPal with the security & privacy of Cryptocurrency

Why can't operating a cryptocurrency wallet be as simple as a PayPal account? Remembering a 256-bit private key is extremely difficult. Password generated keys can be easily brute-forced. Relying on a third-party services to store your keys for you is not only a security but also a privacy risk.

ZeroWallet provides a **non-custodial**, **privacy-centric** and **brute-force resistant** method to recover private keys via **low-entropy passwords**.

ZeroWallet is opensource: http://github.com/amsee01/ZeroWallet


Generate & Recover Your Wallet Below!

Your Private Key

071edf5ab7480c9b0bf77fca54cc475c9
0c19f457b88fd248cba18739d835d2

Your Private Share

8037e20c57b56a03614d5c7c64aa95809
e830cb706e45802a660764ac1d04a36d5b

Decrypt Private Key

test

•••••••

•••••••

Use the fully functional interface to recover your private key with just passwords

1. Visit **app.zerowallet.me**
2. Enter your username and **two** passwords. If you don't have an account, it will be created.
3. Your private key and backup share will be recovered in a matter of seconds.

Note: If you forget your password, the backup share can be used to recover your wallet.

ZeroWallet is a Threshold Multi-Signature Protocol with Zero Knowledge Proofs

(2,3) Secret Sharing

ZeroWallet uses **two** input passwords to generate 3 private key shares:

– **Share 1** = Hash (Pwd 1)
– **Share 2** = Hash (Pwd 2) transformed by Server key
– **Share 3** = Random full-entropy share

Oblivious PRFs

An Oblivious Pseudo-random Function (OPRF) is used to transform the second password by a Server Key. OPRF's beauty lies in that the **client never sees the Server Key** & the **Server never sees the client's password.**

Brute Force Resistant

The Server Key is full entropy. This means that a public brute force attack is of equal difficulty as a full-entropy private key brute force. Server side brute-force is also difficult due to two passwords & slow hashing.

Complete Privacy

Unlike Multi-Sig services like BitGo which need to co-sign your every transaction (and have **censorship potential**), ZeroWallet allows you to simply recover your wallet and carry out transactions independently & privately.

OPAQUE adapted OPRF

The Zero-Knowledge component of this project is the OPRF, which is inspired by the OPAQUE protocol and uses ECC:

1. Client picks a random value r and a password pw.
2. They calculate H(pw), where H represents a one-way hash
3. They use H(pw) to calculate the generator point P on the elliptic curve.
4. They calculate $\alpha = (P)^r$ and send α to the server.
5. The server then calculates $\beta = \alpha^{Ks}$ where Ks is a private server key for each registered user. Ks is randomly generated on user registration and is the only value stored by the server.
6. β is then sent back to the client.
7. The user first calculates m = ModInv(r)and then $rw = \beta^m$.
8. H(pw, β^m) is the randomized output of the OPRF that is used as the second share to generate the user's private key.

To prevent the server from brute-forcing H(*pw*), the hash is multiplied by a large scalar *r*. Similarly, for the user to calculate the value of Ks from β is a hard problem. This prevents either side from cracking the password/server key.

How it Works

Step 1: Client-side hashing and OPRF initialisation

1. The user enters their user name (*usr*), and two password (*pw1*, *pw2*).
2. pw1, pw2, usr are hashed. α is calculated using H(*pw1*) and a random *r*.
3. H(*usr*) and α are transmitted to the server as a POST request.

Step 2: Server-side OPRF

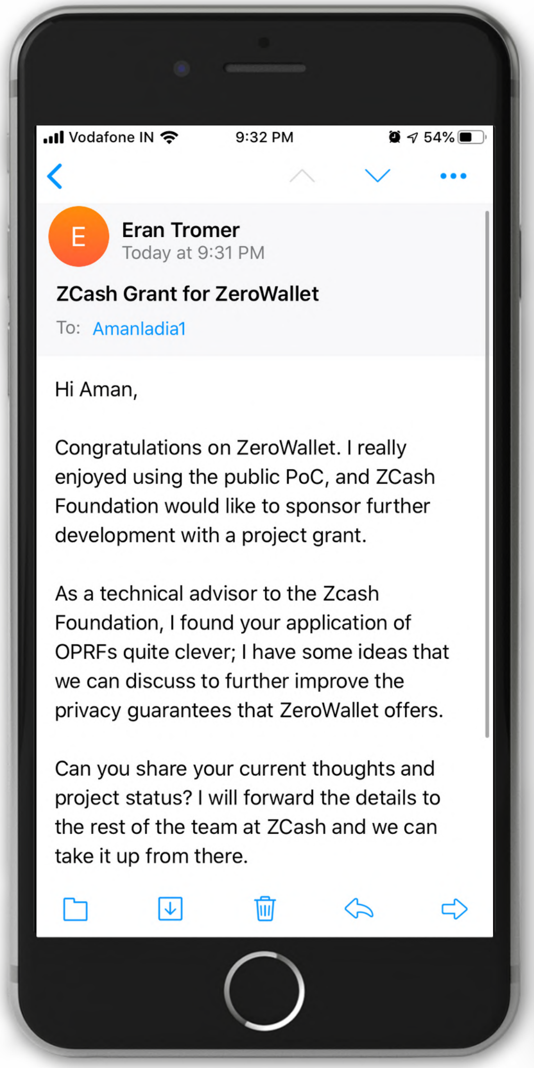
1. Server uses H(usr) to retrieve the user's random *Ks* from a MySQL database. New users: random *Ks* generated and inserted into DB.
2. The server uses the *Ks* to calculate β and send it back as a response.

Step 3: Client-side Calculation and Completion

1. The client obtained β to first calculate m = ModInv(r)and then $rw = \beta^m$. H(*pw1*, *rw*) is the final output of the OPRF.
2. H(*pw2*) and H(*pw1*, *rw*) used as shares for a *(2, 3)* Shamir's Secret Sharing Scheme. Generated secret is hashed to a valid Ethereum priv. key.
3. It uses this secret to generate a third secret share.
4. The priv. key and secret share are output with corresponding bar codes.

ZeroWallet is now nominated for a ZCash Foundation Grant.

Here are some testimonials:



Vodafone IN 9:32 PM 54%

Eran Tromer
Today at 9:31 PM

ZCash Grant for ZeroWallet

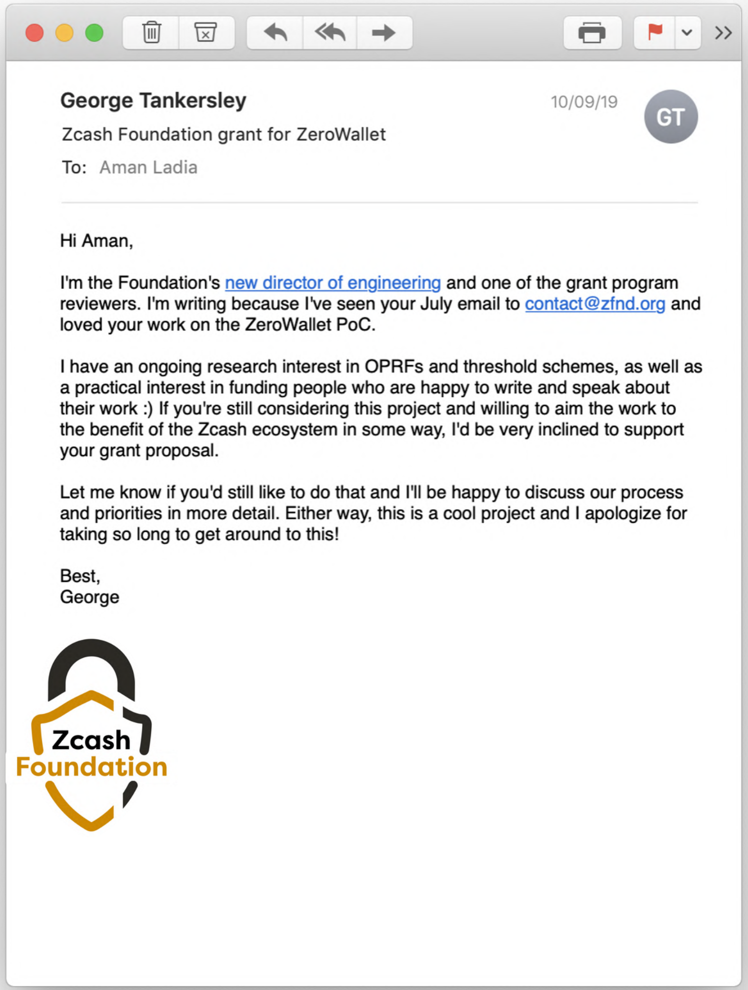
To: Amanladia1

Hi Aman,

Congratulations on ZeroWallet. I really enjoyed using the public PoC, and ZCash Foundation would like to sponsor further development with a project grant.

As a technical advisor to the Zcash Foundation, I found your application of OPRFs quite clever; I have some ideas that we can discuss to further improve the privacy guarantees that ZeroWallet offers.

Can you share your current thoughts and project status? I will forward the details to the rest of the team at ZCash and we can take it up from there.



George Tankersley 10/09/19

Zcash Foundation grant for ZeroWallet

To: Aman Ladia

Hi Aman,

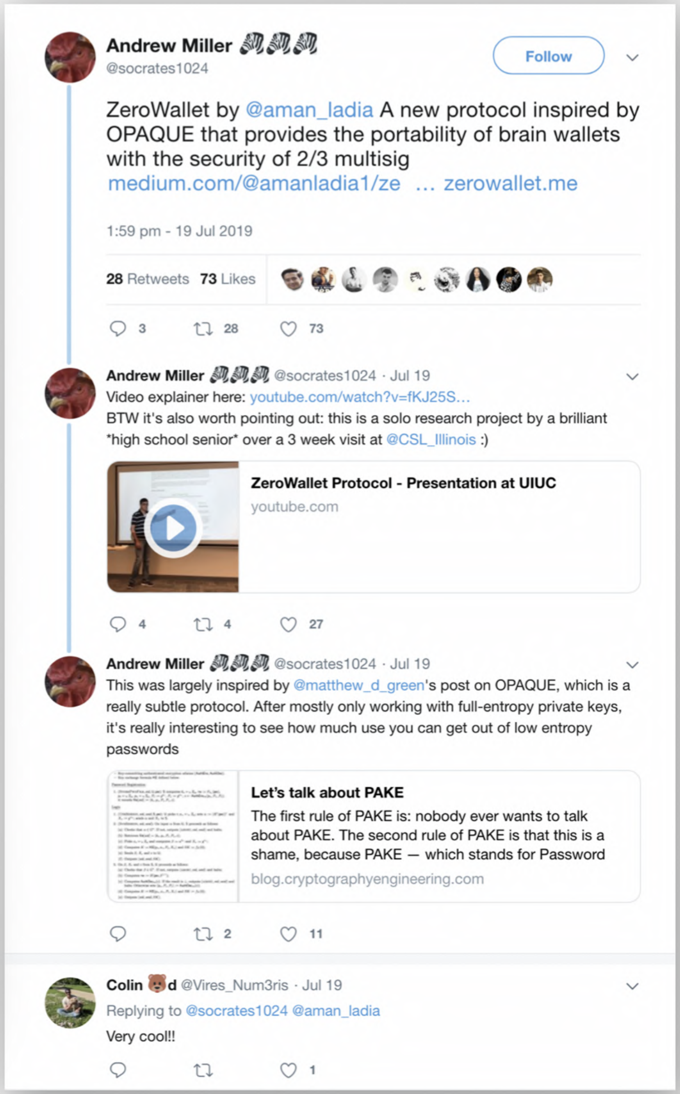
I'm the Foundation's new director of engineering and one of the grant program reviewers. I'm writing because I've seen your July email to contact@zfnd.org and loved your work on the ZeroWallet PoC.

I have an ongoing research interest in OPRFs and threshold schemes, as well as a practical interest in funding people who are happy to write and speak about their work :) If you're still considering this project and willing to aim the work to the benefit of the Zcash ecosystem in some way, I'd be very inclined to support your grant proposal.

Let me know if you'd still like to do that and I'll be happy to discuss our process and priorities in more detail. Either way, this is a cool project and I apologize for taking so long to get around to this!

Best,
George

Zcash Foundation



Andrew Miller @socrates1024

ZeroWallet by @aman_ladia A new protocol inspired by OPAQUE that provides the portability of brain wallets with the security of 2/3 multisig
medium.com/@amanladia1/ze … zerowallet.me

1:59 pm - 19 Jul 2019

28 Retweets 73 Likes

Andrew Miller @socrates1024 · Jul 19
Video explainer here: youtube.com/watch?v=fKJ25S…
BTW it's also worth pointing out: this is a solo research project by a brilliant *high school senior* over a 3 week visit at @CSL_Illinois :)

ZeroWallet Protocol - Presentation at UIUC
youtube.com

Andrew Miller @socrates1024 · Jul 19
This was largely inspired by @matthew_d_green's post on OPAQUE, which is a really subtle protocol. After mostly only working with full-entropy private keys, it's really interesting to see how much use you can get out of low entropy passwords

Let's talk about PAKE
The first rule of PAKE is: nobody ever wants to talk about PAKE. The second rule of PAKE is that this is a shame, because PAKE — which stands for Password
blog.cryptographyengineering.com

Colin @Vires_Num3ris · Jul 19
Replying to @socrates1024 @aman_ladia
Very cool!!

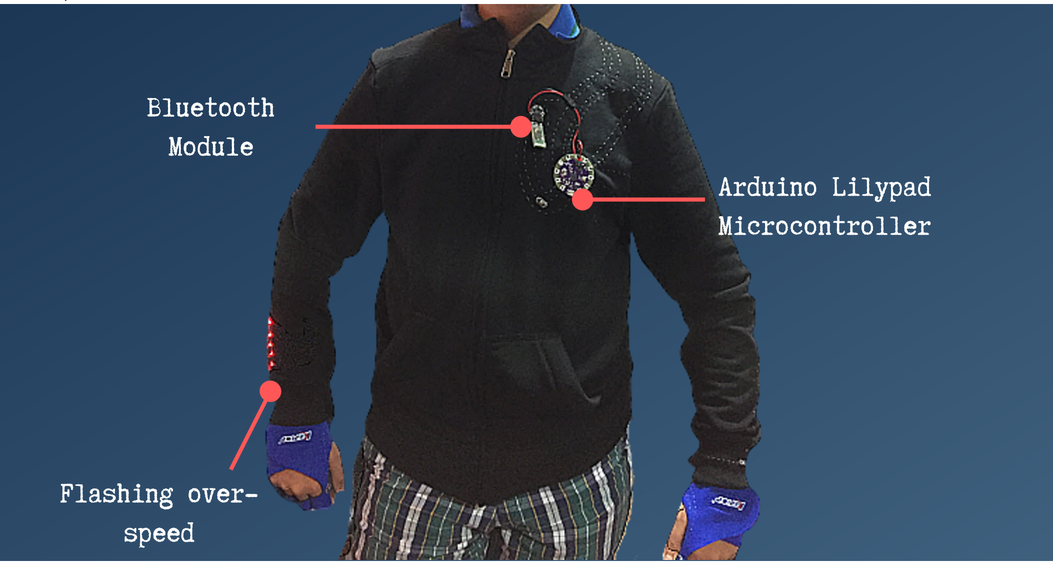
Coverage by IC3 Univ. Research Group: https://initc3.org

Coverage by UIUC CS Lab: https://csl.illinois.edu/news/high-school-senior-develops-new-cryptocurrency-protocol-csl

Jacket Automation for Rider's Safety (JARS)

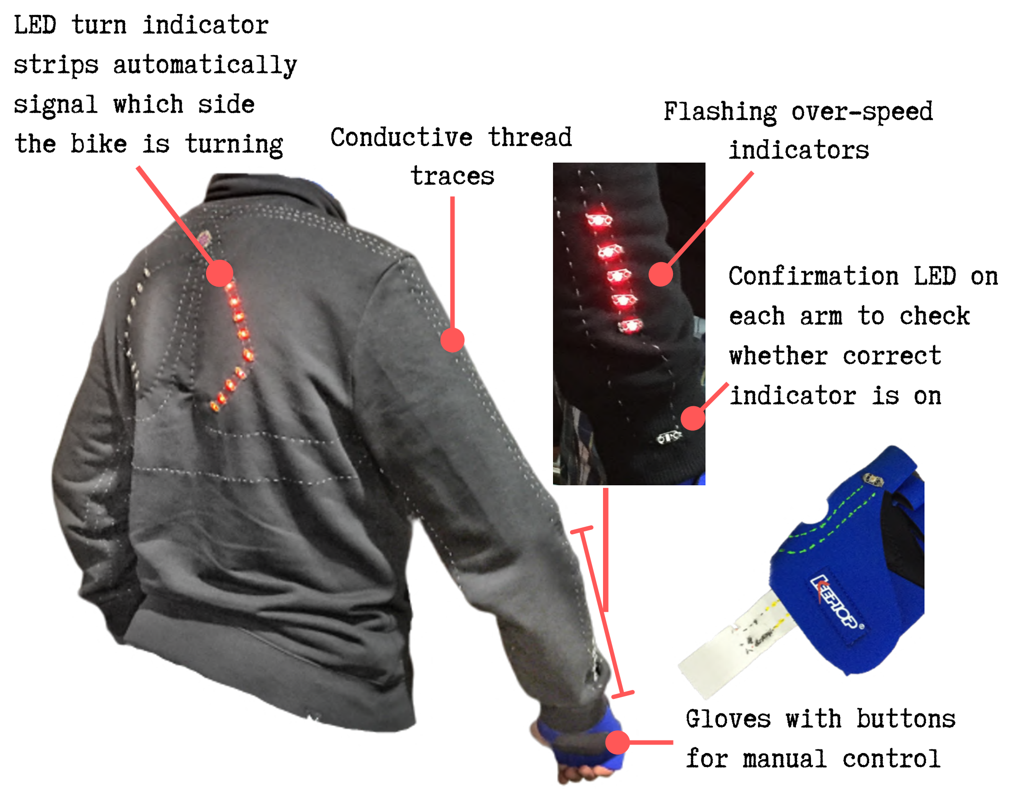
bit.ly/JARSResearch

2016 Onward
IIT & Govt. of India



Overspeeding and Turn Indicator Negligence Kills

65,000 lives are lost to two-wheeler accidents every year. Overspeeding causing accidents is no mystery. But surprisingly, **the next biggest killer is improper use of turn-indicators**. Forgetting to use them or signalling incorrectly is bound to cause an accident. The problem is even worse for cyclists, who are forced to let go of the handle bars and signal a turn with their hand.

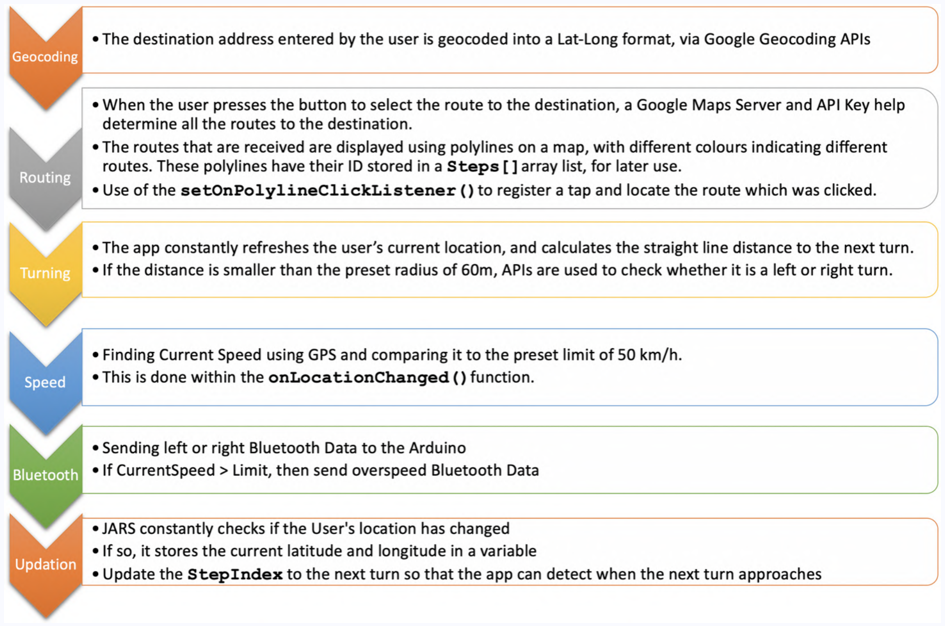

LED turn indicator strips automatically signal which side the bike is turning
Conductive thread traces
Flashing over-speed indicators
Confirmation LED on each arm to check whether correct indicator is on
Gloves with buttons for manual control

Introducing JARS, a smart jacket that improves road safety for bikers.

1. JARS uses the 2 large LED arrows on its back as turn indicators. It automatically signals the correct turn **60m in advance** via GPS & GMaps.
2. Indicator arrows are strategically mounted at the back of the jacket. **Increased height** gives **better visibility** to motorcyclists/bikers.
3. The arms of the jacket contain **over-speed warning lights** that caution the biker to slow down before the situation goes out of hand
4. JARS is **washable** & **completely safe** to wear. Although implemented on a jacket, its design can be **easily adapted** to t-shirts, pullovers etc.

Mobile App Connectivity

JARS makes use of GPS navigation through a paired phone to correctly switch on the turn indicators. The user feeds their destination in the app, selects the preferred Google Maps route and starts driving. The mobile app then pushes data to the Arduino LilyPad via Bluetooth, which executes the physical actions associated with the commands sent by the app.



Geocoding
- The destination address entered by the user is geocoded into a Lat-Long format, via Google Geocoding APIs

Routing
- When the user presses the button to select the route to the destination, a Google Maps Server and API Key help determine all the routes to the destination.
- The routes that are received are displayed using polylines on a map, with different colours indicating different routes. These polylines have their ID stored in a `Steps[]` array list, for later use.
- Use of the `setOnPolylineClickListener()` to register a tap and locate the route which was clicked.

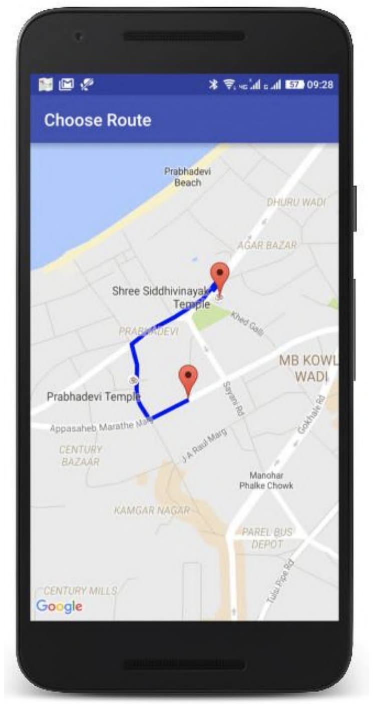
Turning
- The app constantly refreshes the user's current location, and calculates the straight line distance to the next turn.
- If the distance is smaller than the preset radius of 60m, APIs are used to check whether it is a left or right turn.

Speed
- Finding Current Speed using GPS and comparing it to the preset limit of 50 km/h.
- This is done within the `onLocationChanged()` function.

Bluetooth
- Sending left or right Bluetooth Data to the Arduino
- If CurrentSpeed > Limit, then send overspeed Bluetooth Data

Updation
- JARS constantly checks if the User's location has changed
- If so, it stores the current latitude and longitude in a variable
- Update the `StepIndex` to the next turn so that the app can detect when the next turn approaches

Note: Google's Android library lacks support for turn-by-turn navigation, which was needed for this project (way back in 2016). I had to build my own library using GMaps HTML responses and a few half-complete open-source projects on GitHub.



Pick a route on the JARS mobile app and experience automatic turn indicators!

https://github.com/amsee01/JARS

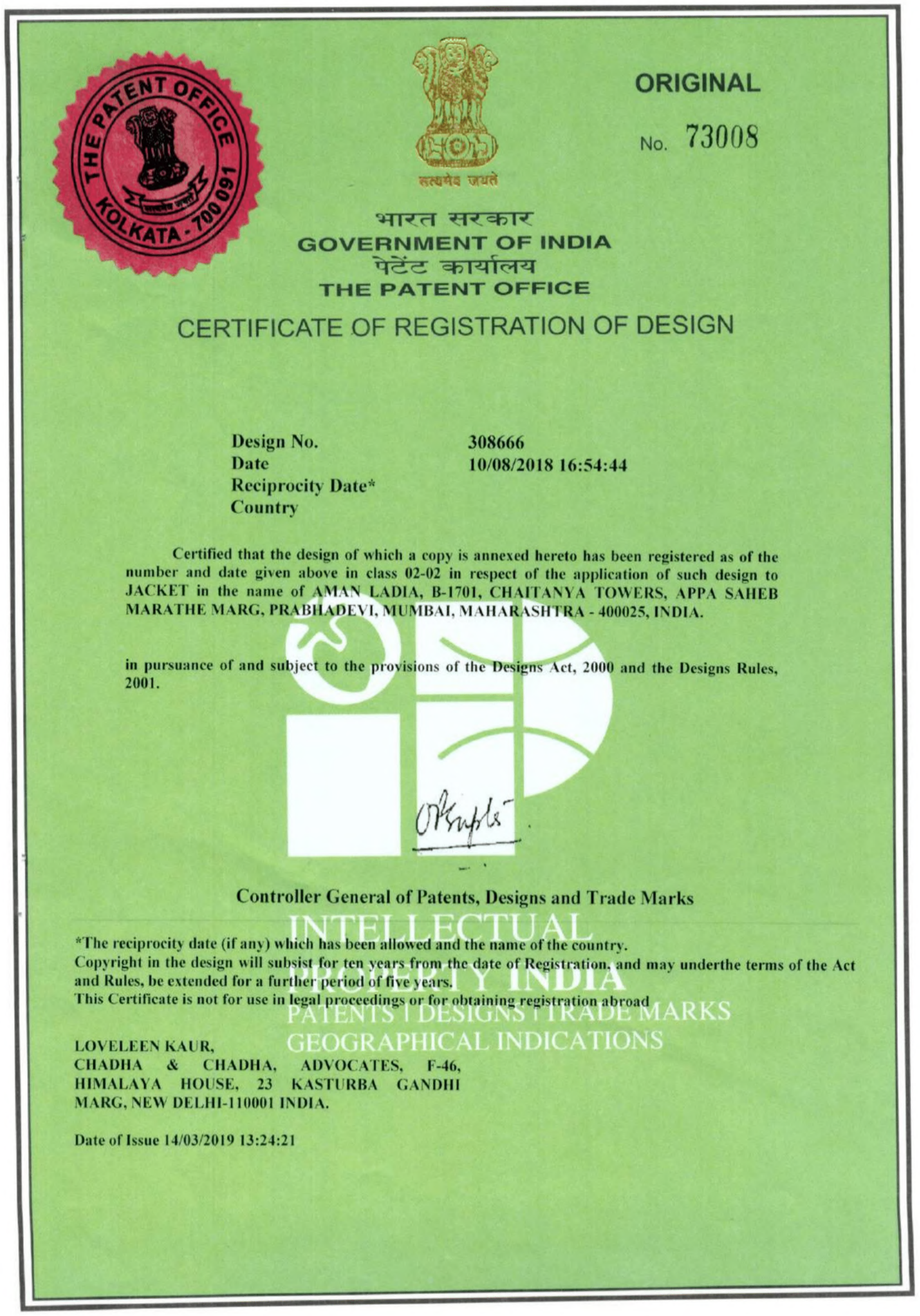
Physical Construction

The base of JARS is an ordinary cloth jacket. All the components of JARS are sewn in and every connection is made with conductive threads. Excluding the removable battery, the jacket is completely safe to wash.

I used a combination of 2-ply silver conductive thread for LEDs and stainless steel thread for higher power electronics like the bluetooth module. The jacket was sewn by hand, and often multiple rounds of stitching had to be performed to reduce the resistance of the circuit. The construction involved lots of trial-and-error with frequent testing.

Components and Costing

JARS Costing (exclusive of jacket price)			
Item	**Cost per unit (₹)**	**Quantity**	**Total Cost (₹)**
LilyPad Arduino USB	345	1	345
LilyPad Red LEDs	10	25	250
Conductive Thread	200	1	200
LilyPad Buttons	23	2	46
HC-05 Bluetooth	184	1	184
LiPo Battery 720 mAh	150	1	150
Final Cost of Components			₹1175 / $16.38



THE PATENT OFFICE KOLKATA - 700 091

सत्यमेव जयते

ORIGINAL

No. 73008

भारत सरकार
GOVERNMENT OF INDIA
पेटेंट कार्यालय
THE PATENT OFFICE

CERTIFICATE OF REGISTRATION OF DESIGN

Design No.	308666
Date	10/08/2018 16:54:44
Reciprocity Date*	
Country	

Certified that the design of which a copy is annexed hereto has been registered as of the number and date given above in class 02-02 in respect of the application of such design to JACKET in the name of AMAN LADIA, B-1701, CHAITANYA TOWERS, APPA SAHEB MARATHE MARG, PRABHADEVI, MUMBAI, MAHARASHTRA - 400025, INDIA.

in pursuance of and subject to the provisions of the Designs Act, 2000 and the Designs Rules, 2001.

[illegible]

Controller General of Patents, Designs and Trade Marks

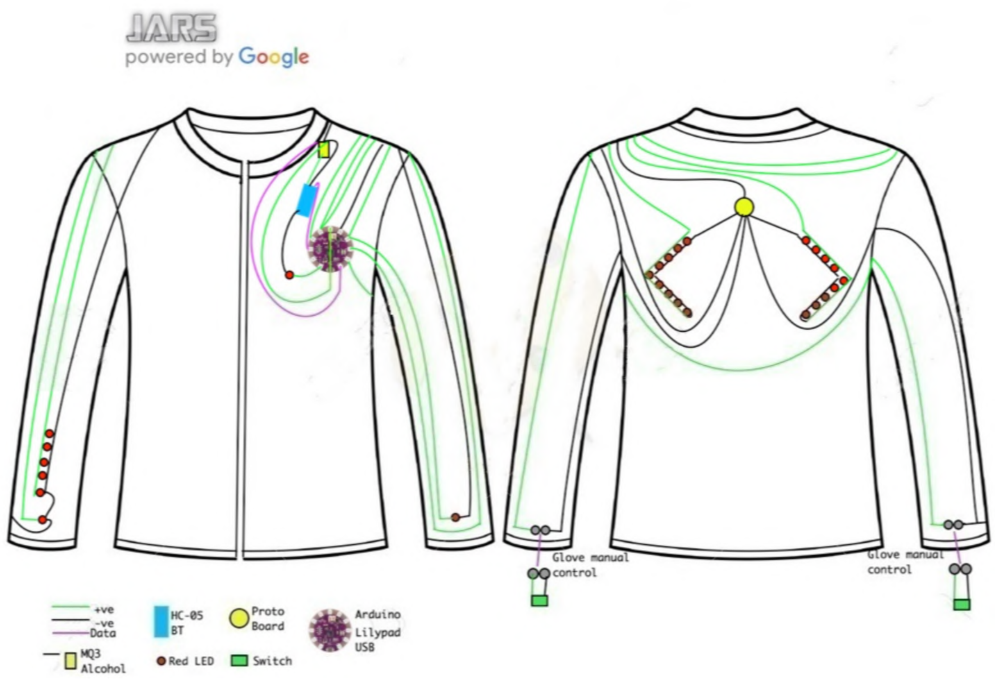
*The reciprocity date (if any) which has been allowed and the name of the country.
Copyright in the design will subsist for ten years from the date of Registration, and may underthe terms of the Act and Rules, be extended for a further period of five years.
This Certificate is not for use in legal proceedings or for obtaining registration abroad

LOVELEEN KAUR,
CHADHA & CHADHA, ADVOCATES, F-46, HIMALAYA HOUSE, 23 KASTURBA GANDHI MARG, NEW DELHI-110001 INDIA.

Date of Issue 14/03/2019 13:24:21

Jacket Wire Traces

As conductive thread is uninsulated, I had to go through multiple revisions to ensure that there were no crossovers in the traces. The final design is below, and **it has received a patent**. Note that in the actual design, the exposed threads were covered with insulating glue for safety reasons.



Ongoing Work

JARS is being minimized to a small patch that can be stuck on the back of any shirt/jacket for added convenience. Moreover, **Machine Learning is being incorporated into the jacket** - instead of just relying on Google Maps, JARS will also study driver patterns with an inbuilt gyroscope, and use that data to accurately predict the next turn, even if the user does not follow GMaps directions.

Pixie: A Hands-Free Home Automation Device for Power Saving

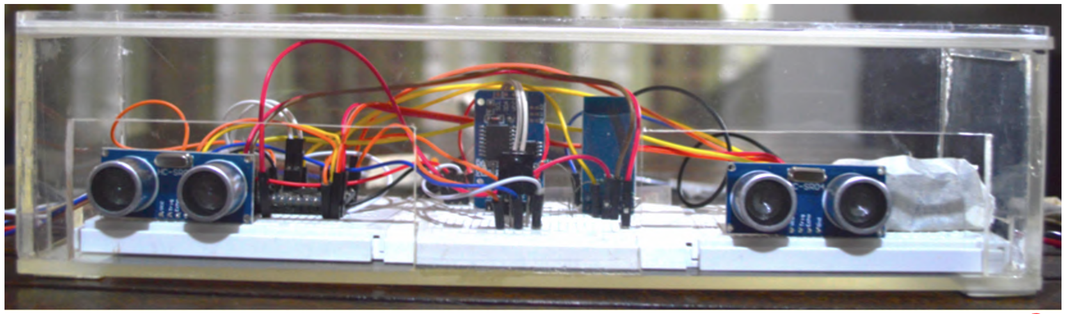
2015 Onward
IEYI & Govt. of India

bit.ly/PixieResearch

Has your five-minute meeting ever turned an hour long?

Did you return to find your lights & A/C left switched on? Meanwhile, 44% of rural households in India **lack complete access to electricity** and nearly **55 million scrape off less than 1 Kilowatt per day** i.e. the same amount that most of our heated/air-conditioned rooms consume in an hour.

I had to drop out of school because there's no electricity at night for me to study



Introducing Pixie: A device that saves every second of power

With so many different excuses for people to leave lights and other appliances on, simply generating awareness on saving power is ineffective.

Instead, Pixie **can adjust appliances automatically**: from switching the lights off as soon as the room is empty, to switching off air-conditioner compressors (whilst still leaving the fan running for circulation), my device bridges the gap between convenience and power saving. Pixie **saves over 1 Kilowatt of electricity per day, lighting up two slum rooms for over 8 hours.**

Pixie outclasses motion sensors

Motion sensors archaic and inefficient in 3 ways:

1. **Delay:** There is typically a 10 min gap between no motion being detected & appliances being turned off, equating to 200W of lost power.
2. **Inconvenience:** The lights switch off if you sit still at a desk while working; you need to flap about to restart lights.
3. **Limited Detection Range** of only 10-15m is a problem.

With Pixie there is no delay, inconvenience or range limit.

Twin Ultrasonic Sensor Algorithm for Automatic Entry and Exit Detection

Pixie is equipped with two ultrasonic sensors. When you enter the room, the **first ultrasonic sensor (say A)** detects your presence *before* the **second one (say B)**, which corresponds to an entry. When you leave the room, the sequence is reversed and Pixie registers it as an exit. This increments/decrement a counting variable, corresponds to the number of people inside a room.
Demo video: https://youtu.be/R-l1PDqJUGo

Smart Lighting Control with Pixie in Manual Mode

Pixie contains a Bluetooth module that allows it to function with the companion app. The app allows users to switch Pixie into a manual mode, where they can toggle the appliances on/off manually. Pixie also features a real time clock (RTC), which enables users to set visual alarms. Pixie can work without a phone as well, over infrared commands issued by several standard TV remotes (https://youtu.be/XA4hYi1cz9k)

Silent Counter – Seamless shift from Manual to Automatic Mode

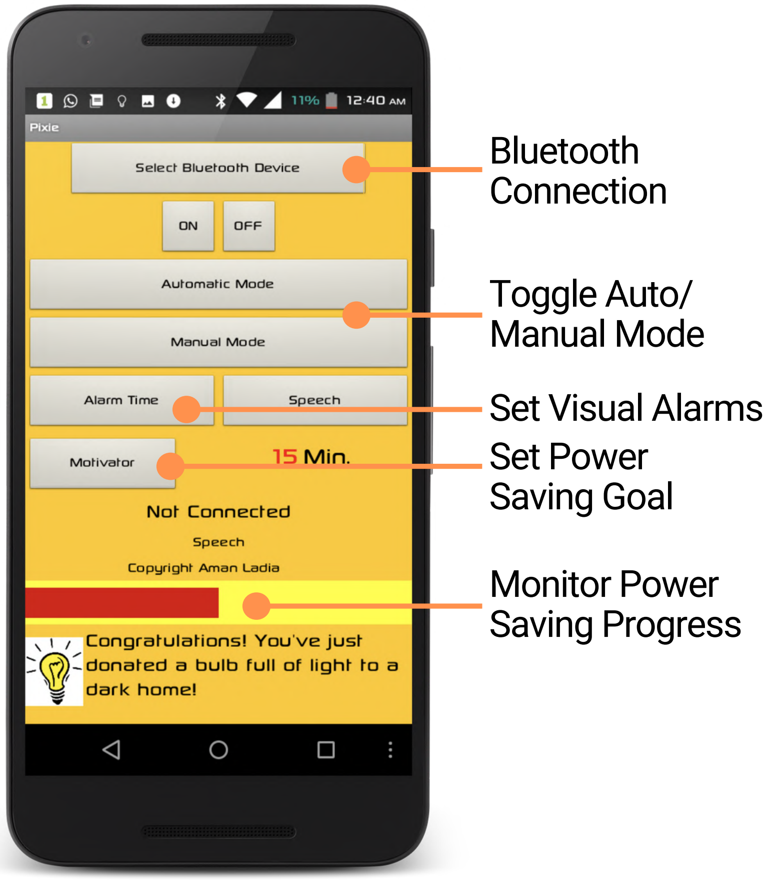
The shift between modes is seamless. Consider that two people enter the room at night and manually switch off the lights. In the night one person leaves. **Even in manual mode, Pixie silently tracks how many people leave or enter the room** without actually turning the lights on or off. The counter will start at 1, which correctly reflects the occupancy.

Complete Functionality: https://youtu.be/Jst23eb4hks

Gamification with Motivation Feature

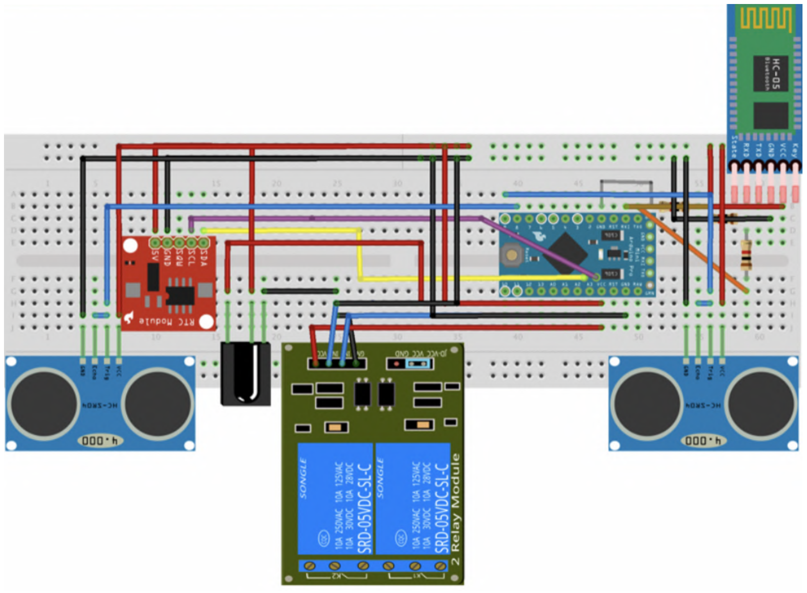
Pixie incorporates an inbuilt motivator feature in its app. Users can set a power saving goal for the day, and view their progress through a progress bar on the app's home screen. This is **also being linked to google games: a power-saving leaderboard** makes energy saving like a game, and the user feels the urge to save maximum electricity to stay at the top.

PIXIE App



Remote

Schematic, Parts List & Costing: Affordable Automation



Pixie Parts & Cost

Part name	Quantity	Total Cost
Large Breadboard	1	₹30.00
Small Breadboard	1	₹25.00
TSOP IR 1738	1	₹15.00
HC-05 Bluetooth Module	1	₹90.00
RTC DS3231	1	₹50.00
1K Resistors	3	₹0.90
Ultrasonic Sensor HC-SR04	2	₹80.00
Arduino Pro Mini	1	₹75.00
IR LED	1	₹5.00
Wires	20	₹20.00
Base Cost (Without Relays)	₹390.9 / $5.43	

Trial Runs and Results

Pixie is installed and running in my room for the past two years. Formally, I have **tested it for 200+ hours** where I made note of every movement into and outside my room. On average, Pixie saved around 1 Kilowatt of power every day through just my lights. With the A/C is included, the saving is about 3 Kilowatts per day for a reasonably careful person who remembers to switch off appliances when leaving the room for regular activities like lunch or dinner.

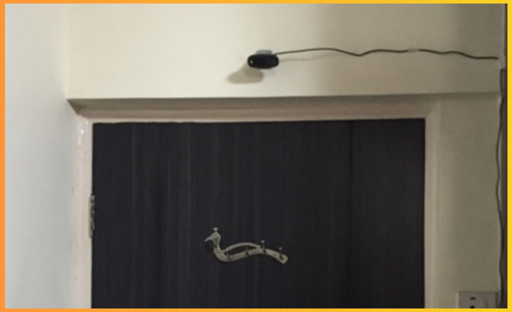
Pixie has also **undergone pilot testing in 4 condos for one year to troubleshoot any problems or bugs.**

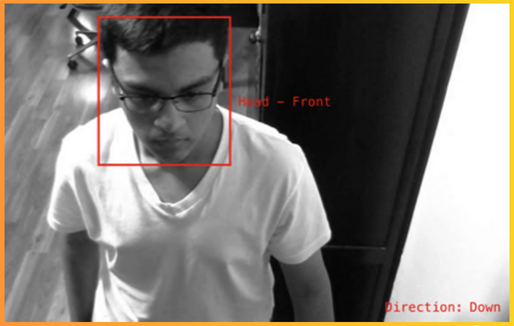
http://bit.ly/PixieResearch

Pixie 2.0: Improved with Computer Vision

To improve Pixie's accuracy and make it workable for large rooms and heavy movement flow, I have designed another model of Pixie that uses a camera along with computer vision and some machine learning to accurately track a room's occupancy.

Pixie 2.0 is still in a beta stage and the code needs fine tuning, but early tests have proved promising.





Mechanical Ramp Attachment for Wheelchairs to Climb Footpaths and Elevated Surfaces

bit.ly/AmanIJMET

2018 Onward

International Journal of Mechanical Engineering and Technology (IJMET)

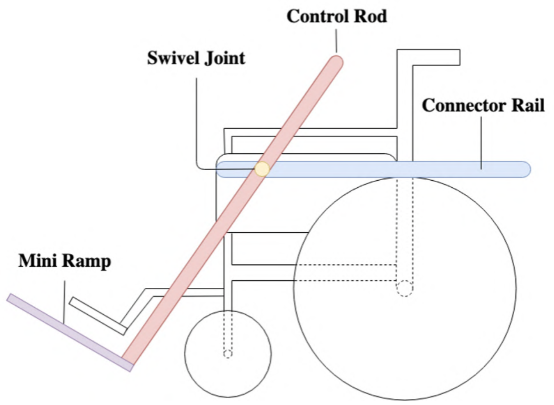
bit.ly/AmanIJMET

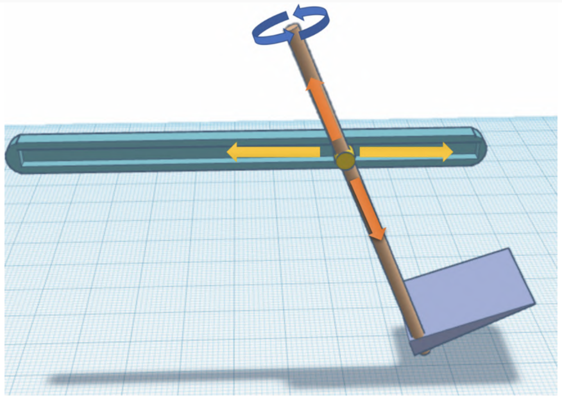
ISSN: 0976-6359

Problem Statement

Wheelchair users should be able to access footpaths and steps through permanent ramps. However, most footpaths in developing countries like India lack wheelchair-access, making handicapped people feel all the more immobile.

As a solution, I designed an **affordable mechanism** for wheelchairs to climb up footpaths in the absence of permanent ramps and without external help. The designed system is **universally adaptable to most wheelchairs**, is built entirely using **mechanical components** and is designed to be **completely self-sufficient**.



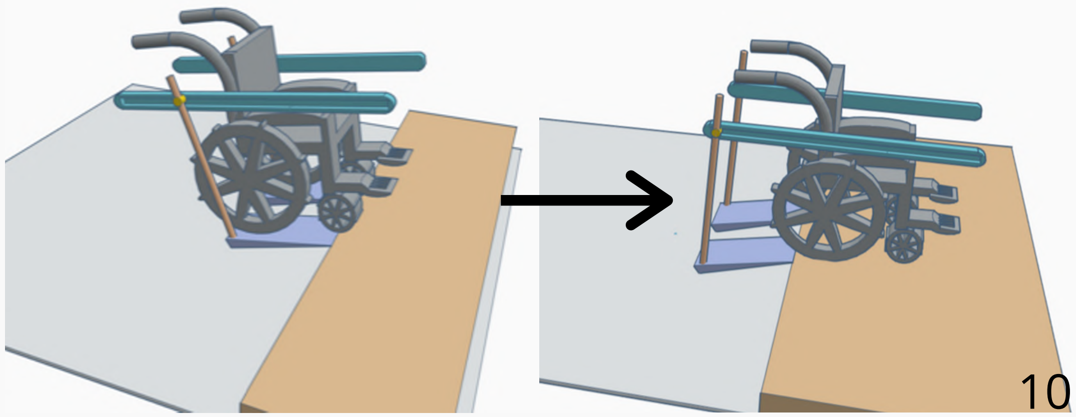


Construction Overview

My implementation consists of a carry-on ramp attachment that stays attached to wheelchairs and uses a mechanism of rods to deploy an inclined plane in front of the wheelchair, thus creating a sloped surface between the road and the pavement. After considering multiple materials, Aluminium Alloy 6061-T6 was chosen for the attachment,

Operation Procedure

1. To deploy the ramp, the handicapped person unlocks the swivel joints & pushes the control rods to align the ramps with the step.
2. They now roll the hind wheels of the wheelchair over the ramps to climb onto the footpath. As they do so, the control rod slips back along the connector rail.
3. Once on the footpath, they rotate the control rods to flip the direction of the mini ramps. They then pull the rods back to the front of the wheelchair, pulling the ramps along.
4. They adjust the rods until the ramps are flush under the foot rests. They then lock the swivel joints.



Mission Rehabilitation of Biodiversity

Ecological engineering to restore urban avian ecosystem

2016 Onward

Indian Teachers Assn. awarded 'Homi Bhabha Young Scientist'

Native birds are endangered in urban environments

The only bird seen freely in Mumbai is the crow. In the absence of any urban bird population surveys, I spent **twelve months** observing my locality's ecosystem, only to find that our **glass buildings**, **open waste disposal sites** and **artificially planted non-native trees** have destroyed the habitat of small birds like sparrows and mynahs. Moreover, **competition from crows** has made perching difficult for these birds.

Engineering 'Biodiversity Domes' – Protected Feeding & Breeding Enclosures

Nut Feeders

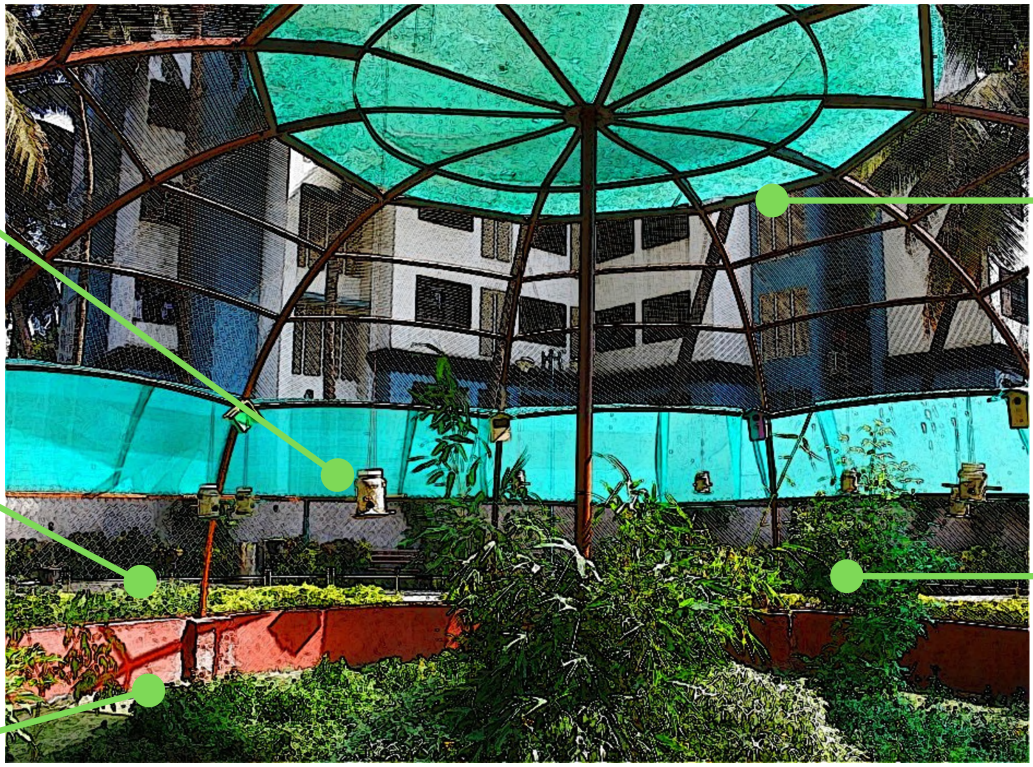
Feeders containing black oil sunflower seeds, safflower and peanuts. Such a blend will provide nutritious food for young sparrows.

Sand Bath

Sand bathing is a part of a bird's preening and plumage maintenance that keeps feathers in good condition.

Water Pool

Water is necessary for sparrows to cool off in the summer heat and for drinking purposes.



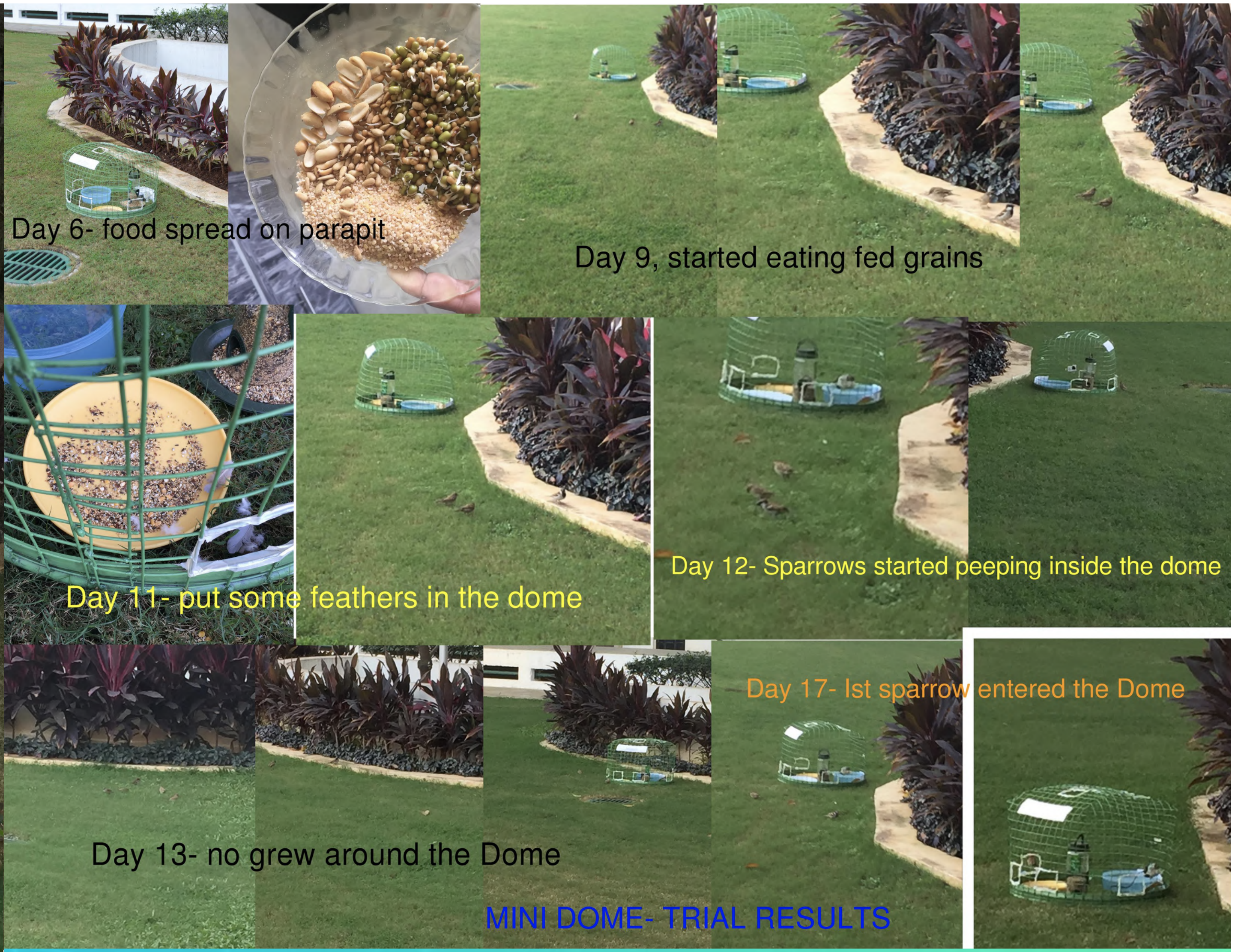
Carefully Sized Holes

8x8 cm holes are large enough for sparrows to enter but too small for crows. This makes biodiversity domes a hotspot for small birds to breed & flourish.

Native Plantation

Planting native flora like *Basil*, *Malabar nut* and *Henna* can help attract sparrows, give them a place to nest & provide food for their growth.

Journey of a Mini Biodiversity Dome Installed in my Locality



Day 6- food spread on parapit

Day 9, started eating fed grains

Day 11- put some feathers in the dome

Day 12- Sparrows started peeping inside the dome

Day 13- no grew around the Dome

Day 17- Ist sparrow entered the Dome

MINI DOME- TRIAL RESULTS

Find the complete 200 page handwritten research & design report at amanladia.com/biodome

Code Appendix – Maker Portfolio

ZeroWallet – Client End JS Code

```
<!DOCTYPE html>

<html lang="en">
<head>
    <title>ZeroWallet</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" type="image/png" href="images/icons/favicon.ico"/>
    <link rel="stylesheet" type="text/css" href="vendor/bootstrap/css/bootstrap.min.css">
    <link rel="stylesheet" type="text/css" href="fonts/font-awesome-4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" type="text/css" href="vendor/animate/animate.css">
    <link rel="stylesheet" type="text/css" href="vendor/css-hamburgers/hamburgers.min.css">
    <link rel="stylesheet" type="text/css" href="vendor/select2/select2.min.css">
    <link rel="stylesheet" type="text/css" href="css/util.css">
    <link rel="stylesheet" type="text/css" href="css/main.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script src="ECCLib.js"></script>
<script src="bundle.js"></script>
<script src='build/forge-sha256.min.js'></script>
<script src="bcmath-min.js" type="text/javascript"></script>
<script src="pdf417-min.js" type="text/javascript"></script>
<script src='decimal.js'></script>
<script src='secrets.min.js'></script>

<script>
$(document).ready(function(){
$("#login").submit(function(event){
  event.preventDefault();

  function h2d(s) {

    function add(x, y) {
        var c = 0, r = [];
        var xx = x.split('').map(Number);
        var yy = y.split('').map(Number);
        while(x.length || y.length) {
            var s = (x.pop() || 0) + (y.pop() || 0) + c;
            r.unshift(s < 10 ? s : s - 10);
            c = s < 10 ? 0 : 1;
        }
        if(c) r.unshift(c);
        return r.join('');
    }

    var dec = '0';
    s.split('').forEach(function(chr) {
        var n = parseInt(chr, 16);
        for(var t = 8; t; t >>= 1) {
            dec = add(dec, dec);
            if(n & t) dec = add(dec, '1');
        }
    });
    return dec;
  }

  function generateBarCode(barcodeVal,element){

    PDF417.init(barcodeVal);
    var barcode = PDF417.getBarcodeArray();
    // block sizes (width and height) in pixels
    var bw = 2;
    var bh = 2;
    var canvas = document.createElement('canvas');
    canvas.width = bw * barcode['num_cols'];
    canvas.height = bh * barcode['num_rows'];


    document.getElementById(element).appendChild(canvas);
    var ctx = canvas.getContext('2d');
    // graph barcode elements
    var y = 0;
    // for each row
      for (var r = 0; r < barcode['num_rows']; ++r) {
          var x = 0;
              // for each column
              for (var c = 0; c < barcode['num_cols']; ++c) {
                  if (barcode['bcode'][r][c] == 1) {
                      ctx.fillRect(x, y, bw, bh);
                  }
                  x += bw;
              }
              y += bh;
          }
  }

// Where the actual code starts


  var usr = $("#username").val();
  var pw2 = $("#pw2").val();
  var hashpw1 = forge_sha256($("#pw1").val());
  var hashpw2 = forge_sha256($("#pw2").val());


  Decimal.set({ toExpPos: 10000 });
  Decimal.set({ precision: 10000 });
  console.log(hashpw1);
  var temp = HashtoPoint(String(h2d(hashpw1)));
  xpw1 = String(temp[0]);
  ypw1 = String(temp[1]);
    console.log("X coordinate of point:" +xpw1)

          var randarray = new Uint32Array(8);
          window.crypto.getRandomValues(randarray);
          r=String(randarray[0]).concat(String(randarray[1]),String(randarray[2]),String(randarray[3]),String(randarray[4]),String(randarra
y[5]),String(randarray[6]),String(randarray[7]));

          rinv = String(ECInverse(r));
          console.log("RANDOM: "+r+" INVERSE:"+rinv);

         temp = ECModExponent(xpw1,r);
         var AlphaX = temp[0];
         var AlphaY = temp[1];
         console.log("Password Coordinates:" +String(xpw1)+" , "+String(ypw1));
         console.log("Randomised Coordinates:"+ String(AlphaX)+" , "+String(AlphaY));
       // /https://piscine-maison-68782.herokuapp.com/
         $.post("https://piscine-maison-68782.herokuapp.com","a=".concat(String(AlphaX),"&u=",String(forge_sha256(usr)),"&t=register"))
           .done(function(data) {

             var Beta = ECModExponent(data, rinv);
             rw = forge_sha256(String(hashpw1).concat(String(Beta[0])));
             console.log("RW: "+rw);

             var shares = [String('801').concat(String(hashpw2)),String('802').concat(String(rw))];
             var privshare = secrets.newShare("03",shares);
             console.log("NEW SHARE: "+privshare);
             var secret = secrets.combine(shares);
             console.log("SECRET: "+secret);

             while(document.getElementById('privkeypic').firstChild != null)
             {
               document.getElementById('privkeypic').removeChild(document.getElementById('privkeypic').firstChild);
             }
                   document.getElementById('bartitle').innerHTML ='Your Private Key';
                   generateBarCode(secret, 'privkeypic');
             var ParNode = document.createElement("p",{});
             var AddrNode = document.createTextNode(secret.substring(0,33));
                   var ParNode2 = document.createElement("p",{});
                   var Addr2Node = document.createTextNode(secret.substring(33));
             ParNode.appendChild(AddrNode);
                   ParNode2.appendChild(Addr2Node);
             document.getElementById('privkeypic').appendChild(ParNode);
                   document.getElementById('privkeypic').appendChild(ParNode2);

```

```
                    while(document.getElementById('sharepic').firstChild != null)
              {
                document.getElementById('sharepic').removeChild(document.getElementById('sharepic').firstChild);
              }
                    document.getElementById('sharetitle').innerHTML ='Your Private Share';
                    generateBarCode(privshare, 'sharepic');
              ParNode = document.createElement("p",{});
              AddrNode = document.createTextNode(privshare.substring(0,33));
                    ParNode2 = document.createElement("p",{});
                    Addr2Node = document.createTextNode(privshare.substring(33));
              ParNode.appendChild(AddrNode);
                    ParNode2.appendChild(Addr2Node);
              document.getElementById('sharepic').appendChild(ParNode);
                    document.getElementById('sharepic').appendChild(ParNode2);

            }
          );
        });
        });
        </script>


        </head>
        <body>

            <div class="limiter">
                <div class="container-login100">
                    <div class="wrap-login100">
                        <div id="PageTitle" class="login100-page-title">
                            Generate & Recover Your Wallet Below!
                            <div class="text-center p-t-1">
                                <a class="txt2" href="http://zerowallet.me" align="right">
                                    New here? Check out how it works.
                                </a>
                            </div>
                        </div>
                        <div id="barcode" class="login100-pic js-tilt" align="center" data-tilt>
                            <span id ="bartitle" class="login100-priv-title">
                            </span>
                            <div id="privkeypic">
                            <img src="images/img-01.png" alt="IMG">
                            </div>
                            <br>
                            <div id="shareprint">
                            <span id ="sharetitle" class="login100-priv-title">
                            </span>
                            <div id="sharepic">
                            </div>
                        </div>
```

```html
                        </div>

                        <form class="login100-form validate-form" id="login" action="" method="post">

                            <span class="login100-form-title">
                                Decrypt Private Key
                            </span>

                            <div class="wrap-input100">
                                <input class="input100" type="text" placeholder="Username" id="username">
                                <span class="focus-input100"></span>
                                <span class="symbol-input100">
                                    <i class="fa fa-user-circle" aria-hidden="true"></i>
                                </span>
                            </div>

                            <div class="wrap-input100">
                                <input class="input100" type="password" placeholder="Password 1" id="pw1">
                                <span class="focus-input100"></span>
                                <span class="symbol-input100">
                                    <i class="fa fa-lock" aria-hidden="true"></i>
                                </span>
                            </div>

                            <div class="wrap-input100">
                                <input class="input100" type="password" placeholder="Password 2" id="pw2">
                                <span class="focus-input100"></span>
                                <span class="symbol-input100">
                                    <i class="fa fa-lock" aria-hidden="true"></i>
                                </span>
                            </div>

                            <div class="container-login100-form-btn">
                                <button class="login100-form-btn">
                                    Decrypt
                                </button>
                            </div>

                            <div class="text-center p-t-10">
                                <a class="txt2" href="http://amanladia.com" align="right">
                                    Created by Aman Ladia
                                </a>
                            </div>
                        </form>
                    </div>
                </div>
            </div>


```

```html



        <script src="vendor/jquery/jquery-3.2.1.min.js"></script>

        <script src="vendor/bootstrap/js/popper.js"></script>
        <script src="vendor/bootstrap/js/bootstrap.min.js"></script>

        <script src="vendor/select2/select2.min.js"></script>

        <script src="vendor/tilt/tilt.jquery.min.js"></script>
        <script >
            $('.js-tilt').tilt({
                scale: 1.1
            })
        </script>

        <script src="js/main.js"></script>

    </body>
    </html>
```

ZeroWallet – Server Node.JS Code

```
var http = require('http');
var ECC = require('./ECCLibServer');
var express=require('express');
var bodyParser = require('body-parser')
const uuidv4 = require('uuid/v4');
const cryptoS = require('crypto');
var cors = require('cors');

var app=express();
app.use(cors());
var urlencodedParser = bodyParser.urlencoded({ extended: false })

var mysql      = require('mysql');
var connection = mysql.createConnection({
  host     : 'HOST',
  user     : 'USER',
  password : 'PASSWORD',
  database : 'DB'
});

connection.connect();

app.post('/', urlencodedParser, function(req,res){
  connection.query("SELECT * FROM Store WHERE user = '".concat(String(req.body.u),"';"), function(error, results, fields){

    if (typeof results[0] === 'undefined') {
      var key = cryptoS.createHash('sha256').update(String(uuidv4()).concat(String(uuidv4())));
      key = key.digest('hex');
      console.log (key);
      var Beta = ECC.ECExponent(String(req.body.a), String(key));

      connection.query("INSERT INTO `Store` (`user`, `store`) VALUES ('".concat(String(req.body.u),"', '",String(key),"');"), function(err
or, results, fields){
          if(error){
            console.log(error);
          }
      });

      res.send(Beta[0]);
    }else {

      var key = cryptoS.createHash('sha256').update(String(results[0].store));
      key = key.digest('hex');
      console.log (key);
      var Beta = ECC.ECExponent(String(req.body.a), String(key));
      res.send(Beta[0]);
    }
  });

});

let port = process.env.PORT;

if (port == null || port == "") {
  port = 3000;
}

var server = app.listen(port, function(){

});
```

ZeroWallet – ECCLibServer Library (adopted from CryptoCoinJS by me)

```
var crypto = require('crypto');

var BigInteger = require('bigi'); //npm install --save bigi@1.1.0
var ecurve = require('ecurve'); //npm install --save ecurve@1.0.0
var cs = require('coinstring'); //npm install --save coinstring@2.0.0
var ecparams = ecurve.getCurveByName('secp256k1',"",""); //remember to remove 2nd and 3rd param from names.js
exports.ECInverse = function (key) {
  var privateKey = new Buffer(key, 'hex')
  var privst1 = BigInteger.fromBuffer(privateKey);
  var privateKeyInv = privst1.modInverse(ecparams.n) ;

  return privateKeyInv;
}

exports.HashtoPoint = function (gx){

  var xpt = new Buffer(String(gx), 'hex')
  var pt = ecparams.pointFromX(true,BigInteger.fromBuffer(xpt));

  return [String(pt.affineX), String(pt.affineY)];

}
exports.ECExponent = function (gx, expo){

  var exponent = BigInteger(String(expo));
  var xpt = BigInteger(String(gx));
  var pt = ecparams.pointFromX(true,xpt);
  var curvePt = pt.multiply(exponent);

  return [String(curvePt.affineX), String(curvePt.affineY)];
}
```

JARS – Jacket Arduino Code

```
#include <SoftwareSerial.h>

int speedInd = 9;
int leftInd = A5;
int rightInd = 2;
int alcoholInd = 20;
int leftButton = A3;
int rightButton = A2;
int stopButton =  44;
int leftcounter = 0;
int rightcounter = 0;
int stopcounter = 0;

int left_state = LOW;
int right_state = LOW;
int stop_state = LOW;

long time =0;
long rtime=0;
long stime=0;

int left_debounce = 10;
int right_debounce = 10;
int lreading;
int rreading;
int stop_debounce = 10;
int stopreading;

char bt;

boolean flag=false;
boolean pressed = false;
boolean rpressed = false;
int stoppressed = false;

boolean leftToggle = false;
boolean lmanual = false;
boolean rmanual = false;

boolean btactive = true;
boolean manualon = false;

boolean lcheck = false;
boolean rcheck = false;

SoftwareSerial mySerial(10, 20);
//SoftwareSerial mySerial(6, 20);

unsigned long previousMillis = 0;        // will store last time LED was updated

unsigned long spreviousMillis = 0;
// constants won't change :
const long interval = 500;
const long sinterval = 200;
int ledState = LOW;
int sledState = LOW;

void setup() {
  // put your setup code here, to run once:
  pinMode(speedInd,OUTPUT);
  pinMode(leftInd,OUTPUT);
  pinMode(rightInd,OUTPUT);
  pinMode(alcoholInd,OUTPUT);
  pinMode(leftButton,INPUT);
  digitalWrite(leftButton,HIGH);
  pinMode(rightButton,INPUT);
  digitalWrite(rightButton,HIGH);

  pinMode(A4,INPUT);
  digitalWrite(A4,HIGH);

  pinMode(stopButton,INPUT);
  digitalWrite(stopButton,HIGH);

  Serial.begin(9600);
  mySerial.begin(9600);
}

void loop() {
    unsigned long currentMillis = millis();
    unsigned long scurrentMillis = millis();
  // put your main code here, to run repeatedly:


  if( mySerial.available() )       // if data is available to read
  {
    ;
  }
  bt = mySerial.read();

  if( bt == 'l' && lmanual == false)
  {
      if (currentMillis - previousMillis >= interval) {
```

```
    // save the last time you blinked the LED
    previousMillis = currentMillis;
    Serial.println(currentMillis - previousMillis);

    // if the LED is off turn it on and vice-versa:
    if (ledState == LOW) {
             ledState = HIGH;
           } else {
             ledState = LOW;
           }

           // set the LED with the ledState of the variable:
           digitalWrite(leftInd, ledState);
           Serial.println("Recd. Left");
         }
         }
         //if( bt == 'r')
         if( bt == 'r' && rmanual == false)
         {
           if (currentMillis - previousMillis >= interval) {
           // save the last time you blinked the LED
           previousMillis = currentMillis;
           Serial.println(currentMillis - previousMillis);

           // if the LED is off turn it on and vice-versa:
           if (ledState == LOW) {
             ledState = HIGH;
           } else {
             ledState = LOW;
           }

           // set the LED with the ledState of the variable:
           digitalWrite(rightInd, ledState);
           Serial.println("INT RIGHT STILL RECD.");
           }


           Serial.println("Recd. Right");

         }


         if( bt == 'e')
         {
           leftInd=A5;
           rightInd=2;
           digitalWrite(leftInd,LOW);
           digitalWrite(rightInd,LOW);

         }

         if(bt == 's')
         {
           if (scurrentMillis - spreviousMillis >= sinterval) {

           spreviousMillis = scurrentMillis;

           if (sledState == LOW) {
             sledState = HIGH;
           } else {
             sledState = LOW;
           }

           // set the LED with the ledState of the variable:
           digitalWrite(speedInd, sledState);
           Serial.println("SPEED RECD.");
         }

         Serial.println("SPEED STILL RECD.");
         }
         if(bt== 'q')
         {
           digitalWrite(speedInd,LOW);
         }


       if(millis() != time)
         {
           lreading = digitalRead(leftButton);

           if(lreading == left_state && leftcounter > 0)
           {
             leftcounter--;
           }
           if(lreading != left_state)
           {
              leftcounter++;
           }

           if(leftcounter >= left_debounce)
           {
             digitalWrite(rightInd,LOW);
             leftInd=A5;
             rightInd=2;
             if(pressed == true)
             {
               pressed=false;
             }
```

```
      else
      {
        pressed=true;
      }
      leftcounter = 0;
      left_state = lreading;
      Serial.println("LEFT BUTTON PRESSED!!!!!!!!!!!!!!!!!!");
      Serial.println(pressed);
      if(left_state == LOW)
      {

        lmanual = !lmanual;
        Serial.println("Toggled "+lmanual);
      }
    }
    time = millis();
  }

  //right button below

  if(millis() != rtime)
  {
    rreading = digitalRead(rightButton);

    if(rreading == right_state && rightcounter > 0)
    {
      rightcounter--;
    }
    if(rreading != right_state)
    {
       rightcounter++;
    }

    if(rightcounter >= right_debounce)
    {
      digitalWrite(leftInd,LOW);
      leftInd=A5;
      rightInd=2;
      if(rpressed == true)
      {
        rpressed=false;
      }
      else
      {
        rpressed=true;
      }
      rightcounter = 0;
      right_state = rreading;
      Serial.println("RIGHT BUTTON PRESSED!!!!!!!!!!!!!!!!!!");
      Serial.println(rpressed);
      if(right_state == LOW)
      {

        rmanual = !rmanual;
        Serial.println("Toggled "+rmanual);
      }
      //leftInd= 20;
      //rightInd=20;
    }
    rtime = millis();
  }

  //Stop code below
  if(millis() != stime)
  {
    stopreading = digitalRead(stopButton);

    if(stopreading == stop_state && stopcounter > 0)
    {
      stopcounter--;
    }
    if(stopreading != stop_state)
    {
       stopcounter++;
    }
    // If the Input has shown the same value for long enough let's switch it
    if(stopcounter >= stop_debounce)
    {

      if(stoppressed == true)
      {
        stoppressed=false;
      }
      else
      {
        stoppressed=true;
      }
      stopcounter = 0;
      stop_state = stopreading;
      Serial.println("STOP BUTTON PRESSED!!!!!!!!!!!!!!!!!!");
      Serial.println(stoppressed);
      if(right_state == LOW)
      {

```

```
                btactive = !btactive;
                Serial.println("Toggled "+btactive);
              }

            }
            stime = millis();
          }


        if(lmanual == true)
        {
          if (currentMillis - previousMillis >= interval) {
            previousMillis = currentMillis;

            if (ledState == LOW) {
              ledState = HIGH;
            } else {
              ledState = LOW;
            }

            digitalWrite(A5, ledState);

          }
          lcheck = true;
        }else if(lmanual == false && lcheck == true)
        {
          digitalWrite(A5,LOW);
          lcheck = false;
        }

        //Right ind code
        if(rmanual == true)
        {
          if (currentMillis - previousMillis >= interval) {
            // save the last time you blinked the LED
            previousMillis = currentMillis;

            if (ledState == LOW) {
              ledState = HIGH;
            } else {
              ledState = LOW;
            }

            digitalWrite(2, ledState);
          }

          rcheck = true;
        }else if(rmanual == false && rcheck == true)
        {
          digitalWrite(2,LOW);
          rcheck=false;
        }

        }
```

JARS – Mobile App Android Code

Part 1: Activity to Choose Destination and Preferred Route

```java
package com.amanladia.jars.sample;

import android.graphics.Color;
import android.os.Bundle;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import com.amanladia.jars.DirectionCallback;
import com.amanladia.jars.GoogleDirection;
import com.amanladia.jars.constant.TransportMode;
import com.amanladia.jars.model.Direction;
import com.amanladia.jars.model.Route;
import com.amanladia.jars.util.DirectionConverter;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.Polyline;
import com.google.android.gms.maps.model.PolylineOptions;

import java.util.ArrayList;

public class AlternativeDirectionActivity extends AppCompatActivity implements OnMapReadyCallback, View.OnClickListener, DirectionCallback
 {
    private Button btnRequestDirection;
    private GoogleMap googleMap;
    private String serverKey = "INSERT KEY HERE";
    private LatLng camera = new LatLng(MainActivity.currentlat, MainActivity.currentlng);
    private LatLng origin = new LatLng(MainActivity.currentlat, MainActivity.currentlng);
    private LatLng destination = new LatLng(MainActivity.lat, MainActivity.lng);

    private String[] colors = {"#7fff7272", "#7f31c7c5", "#7fff8a00"};
    private int blueCol = -16776961;

    public static ArrayList <Polyline> poly = new ArrayList<Polyline>();
    public static ArrayList <Route> sRoute = new ArrayList<Route>();

    public static String polyIndex;
    public static ArrayList <String> storePolId = new ArrayList<String>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_alternative_direction);

        btnRequestDirection = (Button) findViewById(R.id.btn_request_direction);
        btnRequestDirection.setOnClickListener(this);

        ((SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.map)).getMapAsync(this);

    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        this.googleMap = googleMap;
        googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(camera, 15));
        googleMap.setOnPolylineClickListener(new GoogleMap.OnPolylineClickListener()
        {

            @Override
            public void onPolylineClick(Polyline poly)
            {

                Toast.makeText(AlternativeDirectionActivity.this,poly.getId(),Toast.LENGTH_SHORT).show();
                polyIndex = poly.getId();
                poly.setWidth(10);
                poly.setColor(blueCol);
                Snackbar.make(btnRequestDirection,"Route Successfully Selected",Snackbar.LENGTH_SHORT).show();
            }
        });
    }

    @Override
    public void onClick(View v) {
        int id = v.getId();
        if (id == R.id.btn_request_direction) {
            requestDirection();
        }
    }

    public void requestDirection() {
        Snackbar.make(btnRequestDirection, "Direction Requesting...", Snackbar.LENGTH_SHORT).show();
        GoogleDirection.withServerKey(serverKey)
                .from(origin)
                .to(destination)
                .transportMode(TransportMode.DRIVING)
                .alternativeRoute(true)
                .execute(this);
    }

    @Override
    public void onDirectionSuccess(Direction direction, String rawBody) {
        Snackbar.make(btnRequestDirection, "Success with status : " + direction.getStatus(), Snackbar.LENGTH_SHORT).show();
        if (direction.isOK()) {
            googleMap.addMarker(new MarkerOptions().position(origin));
                    googleMap.addMarker(new MarkerOptions().position(destination));

                    storePolId.clear();

                    for (int i = 0; i < direction.getRouteList().size(); i++) {
                        Route route = direction.getRouteList().get(i);
                        sRoute.add(i,route);
                        String color = colors[i % colors.length];
                        ArrayList<LatLng> directionPositionList = route.getLegList().get(0).getDirectionPoint();
                        PolylineOptions opt = new PolylineOptions();
                        opt = DirectionConverter.createPolyline(this, directionPositionList, 5, Color.parseColor(color));
                        poly.add(i, googleMap.addPolyline(opt));
                        poly.get(i).setClickable(true);
                        storePolId.add(i,poly.get(i).getId());


                        //googleMap.addPolyline(DirectionConverter.createPolyline(this, directionPositionList, 5, Color.parseColor(color)))
.setClickable(true);
                    }

                    btnRequestDirection.setVisibility(View.GONE);
                }
            }

            @Override
            public void onDirectionFailure(Throwable t) {
                Snackbar.make(btnRequestDirection, t.getMessage(), Snackbar.LENGTH_SHORT).show();
            }

        }
```

JARS – Mobile App Android Code
Part 2: Activity to Send Commands to Jacket

```
package com.amanladia.jars.sample;

import android.Manifest;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothGattCharacteristic;
import android.bluetooth.BluetoothGattService;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.ServiceConnection;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Criteria;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.net.Uri;
import android.os.IBinder;
import android.support.design.widget.Snackbar;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ExpandableListView;
import android.widget.SimpleExpandableListAdapter;
import android.widget.TextView;
import android.widget.Toast;

import com.amanladia.jars.model.Step;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
```

```java
import java.util.HashMap;
import java.util.List;


public class MainActivity extends AppCompatActivity implements View.OnClickListener, LocationListener {

    private Button btnClick;
    private Button btnCood;
    private Button nextStep;
    public static double lat;
    public static double lng;
    public static TextView cood;
    public static double getLat;
    public static double getLong;

    //current location code below
    private LocationManager locationManager;
    private String provider;
    public TextView latituteField;
    public TextView longitudeField;
    public static double currentlat;
    public static double currentlng;
    public static int manInd=1;
    public static float[] results= new float[1];
    public static Step a;
    public static Step man;
    public static boolean checkInit=false;
    public static boolean inRange= false;
    public static boolean indCheck = false;
    public static float radius = 60;
    public static float distance;

    //BT Send below
    BluetoothAdapter mBluetoothAdapter;
    BluetoothSocket mmSocket;
    BluetoothDevice mmDevice;
    OutputStream mmOutputStream;
    InputStream mmInputStream;
    Thread workerThread;
    byte[] readBuffer;
    int readBufferPosition;
    int counter;
    volatile boolean stopWorker;
    public static boolean isSpeeding = false;

    public static boolean turntaken = false;


    private final static String TAG = MainActivity.class.getSimpleName();

    public static final String EXTRAS_DEVICE_NAME = "DEVICE_NAME";
    public static final String EXTRAS_DEVICE_ADDRESS = "DEVICE_ADDRESS";

    private TextView mConnectionState;
    private TextView mDataField;
            private String mDeviceName;
            private String mDeviceAddress;
            private ExpandableListView mGattServicesList;
            private BluetoothLeService mBluetoothLeService;
            private ArrayList<ArrayList<BluetoothGattCharacteristic>> mGattCharacteristics =
                    new ArrayList<ArrayList<BluetoothGattCharacteristic>>();
            private boolean mConnected = false;
            private BluetoothGattCharacteristic mNotifyCharacteristic;

            private final String LIST_NAME = "NAME";
            private final String LIST_UUID = "UUID";

            // Code to manage Service lifecycle.
            private final ServiceConnection mServiceConnection = new ServiceConnection() {

                @Override
                public void onServiceConnected(ComponentName componentName, IBinder service) {
                    mBluetoothLeService = ((BluetoothLeService.LocalBinder) service).getService();
                    if (!mBluetoothLeService.initialize()) {
                        Log.e(TAG, "Unable to initialize Bluetooth");
                        finish();
                    }
                    // Automatically connects to the device upon successful start-up initialization.
                    mBluetoothLeService.connect(mDeviceAddress);
                }

                @Override
                public void onServiceDisconnected(ComponentName componentName) {
                    mBluetoothLeService = null;
                }
            };

            // Handles various events fired by the Service.
            // ACTION_GATT_CONNECTED: connected to a GATT server.
            // ACTION_GATT_DISCONNECTED: disconnected from a GATT server.
            // ACTION_GATT_SERVICES_DISCOVERED: discovered GATT services.
            // ACTION_DATA_AVAILABLE: received data from the device.  This can be a result of read
            //                        or notification operations.
            private final BroadcastReceiver mGattUpdateReceiver = new BroadcastReceiver() {
                @Override
                public void onReceive(Context context, Intent intent) {
                    final String action = intent.getAction();
                    if (BluetoothLeService.ACTION_GATT_CONNECTED.equals(action)) {
```

```java
                mConnected = true;
                updateConnectionState(R.string.connected);
                invalidateOptionsMenu();
            } else if (BluetoothLeService.ACTION_GATT_DISCONNECTED.equals(action)) {
                mConnected = false;
                updateConnectionState(R.string.disconnected);
                invalidateOptionsMenu();
                clearUI();
            } else if (BluetoothLeService.ACTION_GATT_SERVICES_DISCOVERED.equals(action)) {
                // Show all the supported services and characteristics on the user interface.
                //displayGattServices(mBluetoothLeService.getSupportedGattServices());
            } else if (BluetoothLeService.ACTION_DATA_AVAILABLE.equals(action)) {
                displayData(intent.getStringExtra(BluetoothLeService.EXTRA_DATA));
            }
        }
    };

    // If a given GATT characteristic is selected, check for supported features.  This sample
    // demonstrates 'Read' and 'Notify' features.  See
    // http://d.android.com/reference/android/bluetooth/BluetoothGatt.html for the complete
    // list of supported characteristic features.
    private final ExpandableListView.OnChildClickListener servicesListClickListner =
            new ExpandableListView.OnChildClickListener() {
                @Override
                public boolean onChildClick(ExpandableListView parent, View v, int groupPosition,
                                            int childPosition, long id) {
                    if (mGattCharacteristics != null) {
                        final BluetoothGattCharacteristic characteristic =
                                mGattCharacteristics.get(groupPosition).get(childPosition);
                        final int charaProp = characteristic.getProperties();
                        if ((charaProp | BluetoothGattCharacteristic.PROPERTY_READ) > 0) {
                            // If there is an active notification on a characteristic, clear
                            // it first so it doesn't update the data field on the user interface.
                            if (mNotifyCharacteristic != null) {
                                mBluetoothLeService.setCharacteristicNotification(
                                        mNotifyCharacteristic, false);
                                mNotifyCharacteristic = null;
                            }
                            mBluetoothLeService.readCharacteristic(characteristic);
                        }
                        if ((charaProp | BluetoothGattCharacteristic.PROPERTY_NOTIFY) > 0) {
                            mNotifyCharacteristic = characteristic;
                            mBluetoothLeService.setCharacteristicNotification(
                                    characteristic, true);
                        }
                        return true;
                    }
                    return false;
                }
            };

    private void clearUI() {
        //mGattServicesList.setAdapter((SimpleExpandableListAdapter) null);
        latituteField.setText(R.string.no_data);
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        findViewById(R.id.btn_simple).setOnClickListener(this);
        findViewById(R.id.btn_transit).setOnClickListener(this);
        findViewById(R.id.btn_alternative).setOnClickListener(this);
        btnClick = (Button) findViewById(R.id.button);
        btnClick.setOnClickListener(this);
        btnCood = (Button) findViewById(R.id.button2);
        btnCood.setOnClickListener(this);
        nextStep = (Button) findViewById(R.id.button4);
        nextStep.setOnClickListener(this);

        latituteField = (TextView) findViewById(R.id.textView2);
        longitudeField = (TextView) findViewById(R.id.textView3);


        locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
        // Define the criteria how to select the locatioin provider -> use
        // default
        Criteria criteria = new Criteria();
        provider = locationManager.getBestProvider(criteria, false);
        if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
            return;
        }
        Location location = locationManager.getLastKnownLocation(provider);

        // Initialize the location fields
        if (location != null) {
            System.out.println("Provider " + provider + " has been selected.");
            onLocationChanged(location);
        } else {
            latituteField.setText("Location not available");
            longitudeField.setText("Location not available");
        }
        final Intent intent = getIntent();
        mDeviceName = intent.getStringExtra(EXTRAS_DEVICE_NAME);
        mDeviceAddress = intent.getStringExtra(EXTRAS_DEVICE_ADDRESS);
        Intent gattServiceIntent = new Intent(this, BluetoothLeService.class);
        bindService(gattServiceIntent, mServiceConnection, BIND_AUTO_CREATE);

        }

        @Override
        protected void onResume() {
            super.onResume();
            if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
 && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
                // TODO: Consider calling
                //    ActivityCompat#requestPermissions
                // here to request the missing permissions, and then overriding
                //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
                //                                          int[] grantResults)
                // to handle the case where the user grants the permission. See the documentation
                // for ActivityCompat#requestPermissions for more details.
                return;
            }
            locationManager.requestLocationUpdates(provider, 400, 1, this);

            super.onResume();
            registerReceiver(mGattUpdateReceiver, makeGattUpdateIntentFilter());
            if (mBluetoothLeService != null) {
                final boolean result = mBluetoothLeService.connect(mDeviceAddress);
                Log.d(TAG, "Connect request result=" + result);
            }
        }

        @Override
        protected void onPause() {
            super.onPause();
            unregisterReceiver(mGattUpdateReceiver);
        }

        @Override
        protected void onDestroy() {
            super.onDestroy();
            unbindService(mServiceConnection);
            mBluetoothLeService = null;
        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
            getMenuInflater().inflate(R.menu.gatt_services, menu);
            if (mConnected) {
                menu.findItem(R.id.menu_connect).setVisible(false);
                menu.findItem(R.id.menu_disconnect).setVisible(true);
            } else {
                menu.findItem(R.id.menu_connect).setVisible(true);
                menu.findItem(R.id.menu_disconnect).setVisible(false);
            }
            return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            switch(item.getItemId()) {
                case R.id.menu_connect:
                    mBluetoothLeService.connect(mDeviceAddress);
                    return true;
                case R.id.menu_disconnect:
                    mBluetoothLeService.disconnect();
                    return true;
                case android.R.id.home:
                    onBackPressed();
                    return true;
            }
            return super.onOptionsItemSelected(item);
        }

        private void updateConnectionState(final int resourceId) {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    //mConnectionState.setText(resourceId);
                }
            });
        }

        private void displayData(String data) {
            if (data != null) {
                mDataField.setText(data);
            }
        }

        // Demonstrates how to iterate through the supported GATT Services/Characteristics.
        // In this sample, we populate the data structure that is bound to the ExpandableListView
        // on the UI.
        private void displayGattServices(List<BluetoothGattService> gattServices) {
            if (gattServices == null) return;
            String uuid = null;
            String unknownServiceString = getResources().getString(R.string.unknown_service);
            String unknownCharaString = getResources().getString(R.string.unknown_characteristic);
            ArrayList<HashMap<String, String>> gattServiceData = new ArrayList<HashMap<String, String>>();
            ArrayList<ArrayList<HashMap<String, String>>> gattCharacteristicData
                    = new ArrayList<ArrayList<HashMap<String, String>>>();
            mGattCharacteristics = new ArrayList<ArrayList<BluetoothGattCharacteristic>>();

            // Loops through available GATT Services.
            for (BluetoothGattService gattService : gattServices) {
                HashMap<String, String> currentServiceData = new HashMap<String, String>();
                uuid = gattService.getUuid().toString();
                currentServiceData.put(
                        LIST_NAME, SampleGattAttributes.lookup(uuid, unknownServiceString));
                currentServiceData.put(LIST_UUID, uuid);
                gattServiceData.add(currentServiceData);

                ArrayList<HashMap<String, String>> gattCharacteristicGroupData =
                        new ArrayList<HashMap<String, String>>();
                List<BluetoothGattCharacteristic> gattCharacteristics =
                        gattService.getCharacteristics();
                ArrayList<BluetoothGattCharacteristic> charas =
                        new ArrayList<BluetoothGattCharacteristic>();

                // Loops through available Characteristics.
                for (BluetoothGattCharacteristic gattCharacteristic : gattCharacteristics) {
                    charas.add(gattCharacteristic);
                    HashMap<String, String> currentCharaData = new HashMap<String, String>();
                    uuid = gattCharacteristic.getUuid().toString();
                    currentCharaData.put(
                            LIST_NAME, SampleGattAttributes.lookup(uuid, unknownCharaString));
                    currentCharaData.put(LIST_UUID, uuid);
                    gattCharacteristicGroupData.add(currentCharaData);
                }
                mGattCharacteristics.add(charas);
                gattCharacteristicData.add(gattCharacteristicGroupData);
            }

            SimpleExpandableListAdapter gattServiceAdapter = new SimpleExpandableListAdapter(
                    this,
                    gattServiceData,
                    android.R.layout.simple_expandable_list_item_2,
                    new String[] {LIST_NAME, LIST_UUID},
                    new int[] { android.R.id.text1, android.R.id.text2 },
                    gattCharacteristicData,
                    android.R.layout.simple_expandable_list_item_2,
                    new String[] {LIST_NAME, LIST_UUID},
                    new int[] { android.R.id.text1, android.R.id.text2 }
            );
            mGattServicesList.setAdapter(gattServiceAdapter);
        }

        private static IntentFilter makeGattUpdateIntentFilter() {
            final IntentFilter intentFilter = new IntentFilter();
            intentFilter.addAction(BluetoothLeService.ACTION_GATT_CONNECTED);
            intentFilter.addAction(BluetoothLeService.ACTION_GATT_DISCONNECTED);
            intentFilter.addAction(BluetoothLeService.ACTION_GATT_SERVICES_DISCOVERED);
            intentFilter.addAction(BluetoothLeService.ACTION_DATA_AVAILABLE);
            return intentFilter;
        }

        public void SpeedWrite(){
            if(mBluetoothLeService != null) {
                mBluetoothLeService.writeCustomCharacteristic(0x73);
                Log.e("MainActivity","Written DATA !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");
            }
            else if(mBluetoothLeService==null)
            {
                //Toast.makeText(null,"Null mBT Service!",Toast.LENGTH_SHORT).show();
            }

        }

        public void LeftWrite(){
            if(mBluetoothLeService != null) {
                mBluetoothLeService.writeCustomCharacteristic(0x6c);
                Log.e("MainActivity","Written DATA !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");
            }
            else if(mBluetoothLeService==null)
            {
                Toast.makeText(null,"Null mBT Service!",Toast.LENGTH_SHORT).show();
            }

        }

        public void RightWrite(){
            if(mBluetoothLeService != null) {
                mBluetoothLeService.writeCustomCharacteristic(0x72);
                Log.e("MainActivity","Written DATA !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");
            }
            else if(mBluetoothLeService==null)
            {
                Toast.makeText(null,"Null mBT Service!",Toast.LENGTH_SHORT).show();
            }

        }

        public void TurnOverWrite(){
            if(mBluetoothLeService != null) {
                mBluetoothLeService.writeCustomCharacteristic(0x65);
                Log.e("MainActivity","Written DATA !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");
            }
            else if(mBluetoothLeService==null)
            {
                Toast.makeText(null,"Null mBT Service!",Toast.LENGTH_SHORT).show();
            }

```

```java
        }

        public void NormalSpeedWrite(){
            if(mBluetoothLeService != null) {
                mBluetoothLeService.writeCustomCharacteristic(0x71);
                Log.e("MainActivity","Written DATA !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");
            }
            else if(mBluetoothLeService==null)
            {
                //Toast.makeText(null,"Null mBT Service!",Toast.LENGTH_SHORT).show();
            }

        }

        public void onClickRead(){
            if(mBluetoothLeService != null) {
                mBluetoothLeService.readCustomCharacteristic();
            }
        }


        @Override
        public void onClick(View v) {
            Context context = this;
            EditText input = (EditText) findViewById(R.id.editText);
            String str = input.getText().toString();
            Geocoder gc = new Geocoder(context);
            cood = (TextView) this.findViewById(R.id.textView);
            if (v == btnClick) {
                try {
                    if (gc.isPresent()) {

                        List<Address> list = null;
                        try {
                            list = gc.getFromLocationName(str, 1);
                        } catch (IOException e) {
                            e.printStackTrace();
                        }

                        Address address = list.get(0);

                        lat = address.getLatitude();
                        lng = address.getLongitude();
                        //String strlat= Double.toString(lat);
                        //String strlong= Double.toString(lng);

                    }
                }catch(Exception ex)
                {
```

```java
                Toast.makeText(this,"Destination not found! Please Recheck!",Toast.LENGTH_SHORT).show();
            }


        }

        //code to display latlong
        if (v == btnCood) {
            try
            {
                for(String s:AlternativeDirectionActivity.storePolId)
                {
                    if(s.equals(AlternativeDirectionActivity.polyIndex))
                    {
                        int finalRouteIndex = AlternativeDirectionActivity.storePolId.indexOf(s);
                        SimpleDirectionActivity.leg = AlternativeDirectionActivity.sRoute.get(finalRouteIndex).getLegList().get(0);

                        Toast.makeText(MainActivity.this,"Found at index" + Integer.toString(finalRouteIndex),Toast.LENGTH_SHORT).show();
                    }
                }
            }catch(Exception ex)
            {
                Toast.makeText(MainActivity.this,"Couldn't find index",Toast.LENGTH_SHORT).show();
            }

            checkInit = true;


        }

        if (v == nextStep) {
            //SimpleDirectionActivity.stepInd++;
            //AlternativeDirectionActivity.poly.clear();
            int index;
            for (index =0;index<AlternativeDirectionActivity.poly.size();index++) {
                AlternativeDirectionActivity.poly.remove(index);

            }
            AlternativeDirectionActivity.poly.trimToSize();

            try
            {
                NormalSpeedWrite();
            }catch(Exception ex)
            {
                //Toast.makeText(this,"Failed to send: "+ex.toString(),Toast.LENGTH_SHORT).show();
                Log.e("MainActivity",ex.toString());
            }


        }


        //code for other activities
        int id = v.getId();
        if (id == R.id.btn_simple) {
            goToSimpleDirection();
        } else if (id == R.id.btn_transit) {
            goToTransitDirection();
        } else if (id == R.id.btn_alternative) {
            goToAlternativeDirection();
        }
    }

    public void goToSimpleDirection() {
        openActivity(SimpleDirectionActivity.class);
    }

    public void goToTransitDirection() {
        openActivity(TransitDirectionActivity.class);
    }

    public void goToAlternativeDirection() {
        openActivity(AlternativeDirectionActivity.class);
    }

    public void openActivity(Class<?> cs) {
        startActivity(new Intent(this, cs));
    }

    public void goToSo(View view) {
        goToUrl(SimpleDirectionActivity.maps);
    }

    private void goToUrl(String s) {

        Uri uriUrl = Uri.parse(s);
        Intent launchBrowser = new Intent(Intent.ACTION_VIEW, uriUrl);
        startActivity(launchBrowser);
    }

    @Override
    public void onLocationChanged(Location location) {

        currentlat = location.getLatitude();
        currentlng = location.getLongitude();
        latituteField.setText(Double.toString(currentlat));
```

```java
                longitudeField.setText(Double.toString(currentlng));

                if(checkInit==true) {
                    a = SimpleDirectionActivity.leg.getStepList().get(SimpleDirectionActivity.stepInd);
                    man = SimpleDirectionActivity.leg.getStepList().get(manInd);

                    SimpleDirectionActivity.maneuver = man.getManeuver();

                    getLong = a.getEndLocation().getLongitude();
                    getLat = a.getEndLocation().getLatitude();
                    Location.distanceBetween(getLat, getLong, currentlat, currentlng, results);
                    distance=results[0];

                    if(distance<15)
                    {
                        Toast.makeText(this,"indCheck=true",Toast.LENGTH_SHORT).show();
                        indCheck=true;
                    }

                    if(distance>20 && indCheck)
                    {
                        Toast.makeText(this,"Indicator OFF",Toast.LENGTH_LONG).show();

                        //sendTurnOver();
                        TurnOverWrite();

                        indCheck=false;
                        Toast.makeText(this,"Ind: "+Integer.toString(SimpleDirectionActivity.stepInd)+"Step: "+Integer.toString(SimpleDirec
tionActivity.leg.getStepList().size()),Toast.LENGTH_SHORT).show();
                        if(SimpleDirectionActivity.stepInd<SimpleDirectionActivity.leg.getStepList().size()) {
                            Toast.makeText(this, "updated indexes", Toast.LENGTH_SHORT).show();
                            SimpleDirectionActivity.stepInd++;

                        }
                        if ((manInd+1)<SimpleDirectionActivity.leg.getStepList().size())
                        {
                            manInd++;
                        }
                        else
                        {
                            Snackbar.make(longitudeField,"Route completed",Snackbar.LENGTH_LONG).show();
                            //sendTurnOver();
                            TurnOverWrite();
                        }
                        inRange=false;

                        //update indexes result below

                        a = SimpleDirectionActivity.leg.getStepList().get(SimpleDirectionActivity.stepInd);
                        getLong = a.getEndLocation().getLongitude();
                        getLat = a.getEndLocation().getLatitude();
                        Location.distanceBetween(getLat, getLong, currentlat, currentlng, results);
                        distance=results[0];

                    } else if(distance>20 && !indCheck)
                    {

                        cood.setText(Double.toString(getLat) + " , " + Double.toString(getLong) + "\n" + SimpleDirectionActivity.maneuver + "\n Distance to next turn: " + Float.toString(distance) + "\t manIndex: " + Integer.toString(manInd) + " stepIndex: " + Integer.toString(SimpleDirectionActivity.stepInd));
                    }


                    if(distance<radius)
                    {
                        cood.setText(SimpleDirectionActivity.maneuver+"\n"+Float.toString(distance));
                        for(int i =0;i<5;i++) {
                            if (SimpleDirectionActivity.maneuver.equals("turn-left")||SimpleDirectionActivity.maneuver.equals("uturn-left")) {

                                //sendLeft();
                                LeftWrite();

                            } else if (SimpleDirectionActivity.maneuver.equals("turn-right")||SimpleDirectionActivity.maneuver.equals("uturn-right")) {

                                //sendRight();
                                RightWrite();
                            }

                        }
                        //inRange=true;
                    }
                    //Gmaps link
                    String stpLat = Double.toString(getLat);
                    String stpLng = Double.toString(getLong);
                    SimpleDirectionActivity.maps = "http://maps.google.com/maps?z=12&t=m&q=loc:" + stpLat + "+" + stpLng;
                    //SimpleDirectionActivity.stepInd++;
                    //manInd++;
                }

                //Speedometer code below

                float nCurrentSpeed = location.getSpeed();

                final Snackbar speedsnack = Snackbar.make(longitudeField,"OVER SPEED!",Snackbar.LENGTH_INDEFINITE);

                if(nCurrentSpeed<13.888) {
```

```
                    speedsnack.dismiss();
                    NormalSpeedWrite();
                    //sendNormal();
                }

                if(nCurrentSpeed>13.888) {
                    speedsnack.setAction("Dismiss", new View.OnClickListener() {
                        @Override
                        public void onClick(View v) {
                            speedsnack.dismiss();
                        }
                    });
                    speedsnack.show();
                    //sendSpeed();
                    SpeedWrite();
                    Toast.makeText(this,"Overspeed sent!",Toast.LENGTH_SHORT).show();

                }

                if(nCurrentSpeed<13.888) {
                    speedsnack.dismiss();
                    NormalSpeedWrite();
                    Log.e("Main Activity","Sent NORMAL SPEED !!!!!!!!");
                    Log.e("Main Activity","Sent NORMAL SPEED !!!!!!!!");
                    Log.e("Main Activity","Sent NORMAL SPEED !!!!!!!!");
                    Log.e("Main Activity","Sent NORMAL SPEED !!!!!!!!");
                    Log.e("Main Activity","Sent NORMAL SPEED !!!!!!!!");
                    //sendNormal();
                }

            }

            @Override
            public void onStatusChanged(String provider, int status, Bundle extras) {

            }

            @Override
            public void onProviderEnabled(String provider) {
                Toast.makeText(this, "Enabled new provider " + provider,
                        Toast.LENGTH_SHORT).show();

            }

            @Override
            public void onProviderDisabled(String provider) {
                Toast.makeText(this, "Disabled provider " + provider,
                        Toast.LENGTH_SHORT).show();
            }
```

```


        }
```

Pixie Arduino Code

```
#include "Wire.h"
#define DS3231_I2C_ADDRESS 0x68
byte byteReadh;
byte byteReadm;
byte byteReadr;
int check=0;
int checkl=0;
int checkc=0;
int alarmh;
int alarmh1;
int alarmm;
int alarmm1;
int alarmm2;
int alarmam;
int checkdisp=0;
int motivate=0;
int timea=0;
int timeb=0;
int checktime=0;
int globehour=0;
int globemin=0;
char bt;
// Convert normal decimal numbers to binary coded decimal
byte decToBcd(byte val)
{
  return( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
  return( (val/16*10) + (val%16) );
}
//replace here


//The RTC and OLED setup ends here

//the autodetect setup starts from below
#include <NewPing.h>
#include <IRremote.h>

int us1Pin = 8;
int us2Pin = 9;
int relayPin = 13;

#define MAX_DISTANCE 200
#define OK_POS1 0x499B750A
#define OK_POS2  0x9E856ADE
#define RECV_PIN 10
#define OK_ON 0x7D03709
#define OK_OFF 0x7AD7A7D9

byte ledState;
IRrecv irrecv(RECV_PIN);
decode_results results;
boolean power_state = LOW;


class USSensor {
public:
  USSensor(int trigPin, int echoPin, String aName) {
    trig = trigPin;
    echo = echoPin;
    sonar = new NewPing(trigPin, echoPin, MAX_DISTANCE);
    isActivated = false;
    name = aName;
    time = 0;
  }

  void update() {
    updateSensor();
    if (value < 30) {
      setActive(true);
    } else {
      if (millis() - time > 500) {
        setActive(false);
      }
    }
  }

  void reset() {
    setActive(false);
    time = 0;
  }

  void setActive(boolean state) {
    if (!isActivated && state) {
      time = millis();
    }
    if (isActivated != state) {
      isActivated = state;
      Serial.print(name + ": ");
      Serial.println(state);
    }
  }

  boolean wasActive() {
    return (time != 0) && !isActivated;
  }

         int lastActive() {
           return millis() - time;
         }

       private:
         int trig;
         int echo;
         int led;
         String name;
         boolean isActivated;
         unsigned long time;
         float value;
         NewPing *sonar;

         void updateSensor() {

           delay(25);
           unsigned int uS = sonar->ping();
           value = uS / US_ROUNDTRIP_CM;

       //    delay(5);
       //    pinMode(trig, OUTPUT);
       //    digitalWrite(trig, LOW);
       //    delayMicroseconds(50);
       //    digitalWrite(trig, HIGH);
       //    delayMicroseconds(200);
       //    digitalWrite(trig, LOW);
       //    pinMode(echo, INPUT);
       //    int pulseLen = pulseIn(echo, HIGH);
       //    float currentValue = pulseLen / 29.387 /2; // [voloshyn] convert into cm
       //    if (value == 0.0f) value = currentValue;
       //    else value = (value + currentValue) / 2;
       //    value = pulseLen / 29.387 /2; // [voloshyn] convert into cm
         }
       };

       int peopleInRoom = 0;
       USSensor *A;
       USSensor *B;

       void setup() {
         irrecv.enableIRIn(); // Start the receiver
         Serial.begin(9600);
         A = new USSensor(us1Pin, us1Pin, "A");
         B = new USSensor(us2Pin, us2Pin, "B");
         pinMode(relayPin, OUTPUT);
         Wire.begin();
          // set the initial time here:
         // DS3231 seconds, minutes, hours, day, date, month, year
         //setDS3231time(30,25,25,2,2,6,15);
       }


       void setDS3231time(byte second, byte minute, byte hour, byte dayOfWeek, byte
       dayOfMonth, byte month, byte year)
       {
         // sets time and date data to DS3231
         Wire.beginTransmission(DS3231_I2C_ADDRESS);
         Wire.write(0); // set next input to start at the seconds register
         Wire.write(decToBcd(second)); // set seconds
         Wire.write(decToBcd(minute)); // set minutes
         Wire.write(decToBcd(hour)); // set hours
         Wire.write(decToBcd(dayOfWeek)); // set day of week (1=Sunday, 7=Saturday)
         Wire.write(decToBcd(dayOfMonth)); // set date (1 to 31)
         Wire.write(decToBcd(month)); // set month
         Wire.write(decToBcd(year)); // set year (0 to 99)
         Wire.endTransmission();
       }
       void readDS3231time(byte *second,
       byte *minute,
       byte *hour,
       byte *dayOfWeek,
       byte *dayOfMonth,
       byte *month,
       byte *year)
       {
         Wire.beginTransmission(DS3231_I2C_ADDRESS);
         Wire.write(0); // set DS3231 register pointer to 00h
         Wire.endTransmission();
         Wire.requestFrom(DS3231_I2C_ADDRESS, 7);
         // request seven bytes of data from DS3231 starting from register 00h
         *second = bcdToDec(Wire.read() & 0x7f);
         *minute = bcdToDec(Wire.read());
         *hour = bcdToDec(Wire.read() & 0x3f);
         *dayOfWeek = bcdToDec(Wire.read());
  *dayOfMonth = bcdToDec(Wire.read());
  *month = bcdToDec(Wire.read());
  *year = bcdToDec(Wire.read());
}

void displayTime()
{
  byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
  // retrieve data from DS3231
  readDS3231time(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month,
  &year);

globehour = hour;
globemin = minute;

 /* if(second > 25 && second < 27){
  // send it to the serial monitor
 Serial.print(hour, DEC);
  // convert the byte variable to a decimal number when displayed
  Serial.print(":");
  if (minute<10)
  {
    Serial.print("0");
  }
  Serial.print(minute, DEC);
  Serial.print(":");
  if (second<10)
  {
    Serial.print("0");
  }
  Serial.println(second, DEC);
  }*/
  checkdisp=minute;
  if (hour==alarmh){
    if(minute==alarmm){
      Serial.println("Alarm!");
      digitalWrite(13,HIGH);

    }
  }
}


void loop() {
  byte byteReadh, byteReadma, byteReadmb, byteReadam, byteReadh1;
     if( Serial.available() )       // if data is available to read
  {
    ;
  }
  bt = Serial.read();         // read it and store it in 'val'

  if( bt == 'A' )               // if 'a' was received led 2 is switched off
  {
    digitalWrite(13, HIGH);    // turn Off pin 2
  }

  if( bt == 'a' )               // if 'A' was received led 2 on
  {
    digitalWrite(13, LOW);  // turn ON pin 2a
  }
  if( bt == 'b')
  {
    relayPin=13;
  }

  if(bt == 'c')
  {
    relayPin=16;
  }

  if(bt == 'h')
  {
  // byteReadh= Serial.read();
  //  alarmh=byteReadh;
    alarmh= Serial.read();
    Serial.println(alarmh);
  }


  if( bt == 'm' )
  {
    Serial.println(motivate);
  }

  if(bt == 'i')
  {
  // byteReadma= Serial.read();
    //alarmm1=byteReadma;
    alarmm= Serial.read();
    Serial.println(alarmm);
```

```
        }

      //Serial.println(bt);


       if (irrecv.decode(&results)) {  //If IR receive results are detected
         Serial.println(results.value, HEX);
          switch (results.value) {

           //Power
          case OK_POS1:
            //   Serial.println("Power");
               digitalWrite(13,LOW);
             // turn LED ON

          break;

           case OK_POS2:
            //   Serial.println("Power");
               digitalWrite(13, HIGH);  // turn LED ON

          break;

          case OK_ON:
            //   Serial.println("Power");
               relayPin=16;  // turn LED ON

          break;

          case OK_OFF:
            //   Serial.println("Power");
               relayPin=13;  // turn LED ON

          break;


          }

          irrecv.resume(); // Receive the next value
        }



          A->update();
          B->update();

          if (A->wasActive() && B->wasActive()) {
            int a_time = A->lastActive();
             int b_time = B->lastActive();
             if (a_time < 5000 && b_time < 5000) {
               if (a_time > b_time) {
                 peopleInRoom++;
               } else {
                 peopleInRoom--;
               }
               Serial.print("People in room: ");
               Serial.println(peopleInRoom);
             }
             A->reset();
             B->reset();
           }
          displayTime();
          byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
         // retrieve data from DS3231
         readDS3231time(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month,
         &year);
           if (peopleInRoom < 0) { // [voloshyn] in case when someone was already in the room when system was activated
             peopleInRoom = 0;
           }

           if (peopleInRoom > 0) {
             digitalWrite(relayPin, HIGH);
             if (checktime==1){
             timea = globehour*60+globemin;
             Serial.println(timea);
             motivate = motivate + timea - timeb;
             }
             checktime = 0;

           } else {
             digitalWrite(relayPin, LOW);
             if (checktime==0){
             delay (100);
             timeb = globehour*60+globemin;
             Serial.println(timeb);
             }
             checktime=1;
           }

       }
```