

Weekly Technical Journal

Senior Project Design 1

Vladimir Nava (C856A927)

Week 9/5 – 9/11

Shared our project ideas amongst ourselves, narrowing it down to just four ideas.

Held an anonymous group vote for the four projects in which we decided upon just two ideas.

The two project ideas are:

Bus System App – Develop software for a bus system that tells when and where traffic maybe at, a queue in which people who use the software can line up for. The project would focus on the time and efficiency for the buses here in Wichita.

Agriculture Sensor System – Utilize sensors to gather information about plants

Learned about the different IoT's used for project ideas (NFC, Hydro Sensor, and Barcode software).

Week 9/12 – 9/18

This week we focused on shifting the project towards school buses and looking to ask a specific school to work with.

Decided on the possible language we will use for the app going forward.

Began to learn Swift by familiarizing myself with the syntax.

Added possible idea of weather updates.

Researched app tracking tools with SDKs:

Google Analytics

Firebase Analytics

I also researched GPS tracking methods that the team can potentially use.

For now the plan is to use the school bus drivers phone as the dedicated GPS tracking device.

Week 9/19 – 9/25

The focus this week was on working on planning the project out.

We chose to use google docs to each work on two sections of the project planning paper.

Any changes made can be done so in real time.

Swift syntax is like C++ in that it has variables, functions, loops, and classes (OOP).

The first thing I figured out was how to get Swift to work on a windows machine.

A toolchain provided by <https://swift.org/blog/swift-on-windows/> makes it possible to program with Swift on Windows 10.

Writing code is working as expected, but compiling is still an issue.

Going to try to get Swift running on WSL (Windows Subsystem for Linux) through Ubuntu 18.04 next.

Week 9/26 – 10/2

Continuing to learn the Swift language.

Almost completed all courses on codecademy.

I figured out a way to run Swift using WSL.

This is used mainly for practice.

Research was conducted this week with regards to requirements.

My job was to gather patent/prior art findings and market findings.

I learned how to search for patents using uspto.gov

I learned about the “Seven Step Strategy” for U.S. patents

The information gathered so far shows that the idea is viable without much trouble from patents.

Week 10/3 – 10/9

Research was finalized this week and compiled into a presentation.

Practice was done so that the final presentation for the midterm would be done so in a concise manner.

Swift coding general learning was finished this week.

The requirements presentation was formatted and presented.

We finalized our decision on making swift the language for our project.

This was done after analyzing the other languages for developing iOS apps (subject-C and C#)

The reason we chose swift was due to it being made specifically for iOS devices along with an easy to learn syntax and a fast development process when compared to other languages. Based on the people we shall be targeting the app towards; C# will not be needed as the app will strictly be an iOS app only.

Week 10/10 – 10/16

The design of the app involves tracking the school bus with updates occurring on a timer. The information stored here will not be encrypted as no information will be displayed from the bus driver or students. If someone were to maliciously tamper with the tracking system, all they would see is the bus routes and nothing more.

For now, the idea is that data will be stored on a raspberry pi 3 server which will be tested for a working prototype.

The original plan was to use SQL to create a database for storing any data we need.

This however overcomplicated the process as not much data will need to be stored.

A simple text document to hold all the necessary values as a much simpler design that fits the criteria for holding data.

The plan is to test this first and move the project forward as necessary.

The server will be pinged when new data has arrived, stored in a text document.

The aforementioned information will be further improved in next week's team meeting.

Week 10/17 – 10/23

The team has discussed how the GPS tracking method should work moving forward.

Rather than rely on the bus drivers' phone as a GPS tracker, we decided on using a dedicated GPS device that will be always on the bus.

This removes the restrictions that tracking by phones brings including regulations.

There are two options that can work for this project.

The first is creating the device ourselves using a microcontroller, GPS module, GSM module, and power supply.

The second option is to buy a GPS tracker off the market.

Both options have pros and cons, the team will weigh these and decide depending on the requirements the school will give us.

The important standard that must be followed here for GPS data is using NMEA format. This is a standard by the National Marine Electronics Association.

I have also learned how GPS modules work along with how this data can be transmitted to a dedicated server.

The process involves using a microcontroller or microcomputer along with Wi-Fi capabilities.

A bit of programming is needed here as well as script writing if the process is to be automated.

The consensus surrounding this is if you wish to track a vehicle, the better alternative is to buy a GPS tracker.

Buying a GPS tracker means paying an upfront price along with a monthly fee for the service.

For this semester, the team has decided on making a GPS tracker via a raspberry pi. The tracker will be stationary and will not have GSM capabilities. This GPS tracker will serve the purpose of getting a better understanding of how GPS data works and how the team can interlink it with a server along with a mobile app.

Depending on the requirements, the tracker can be improved in the second semester to accommodate for mobility as well as power consumption. This is subject to change as buying a tracker or another form of GPS tracking is still an option.

The raspberry pi is a 3rd generation model b with the specifications (courtesy of raspberry pi):

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
- 100 Base Ethernet
- 40-pin extended GPIO
- 4 USB 2 ports
- 4 Pole stereo output and composite video port
- Full size HDMI
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A

The GPS module is a GPS Module GPS NEO-6M with specifications(courtesy of amazon):

With IPEX antenna interface, the default distribution of active antenna, can be quickly positioned Operating voltage: 3.6V-5V (or direct usb power supply) Operating baud rate: 9600 (can be modified) Onboard rechargeable button battery Onboard E2PROM can save parameter data NEMA output format is compatible with NEO-6M Size: 27.6mm * 26.6mm can be inserted or selected patch (with positioning holes)

Week 10/24 – 10/30

The team has met with the school where we discussed what solutions that a school bus app brings.

The school has agreed that if buying a GPS tracker is necessary, they are willing to do it.

More research needs to be done to figure out how to extract GPS data from a GPS tracker into a cloud server.

Another issue is how to display that data on a map via the mobile app.

There are two services that are of interest: AWS and Firebase.

Firebase created by google, helps in the creation of mobile apps for both iOS and Android.

AWS is a cloud storage service provided by Amazon.

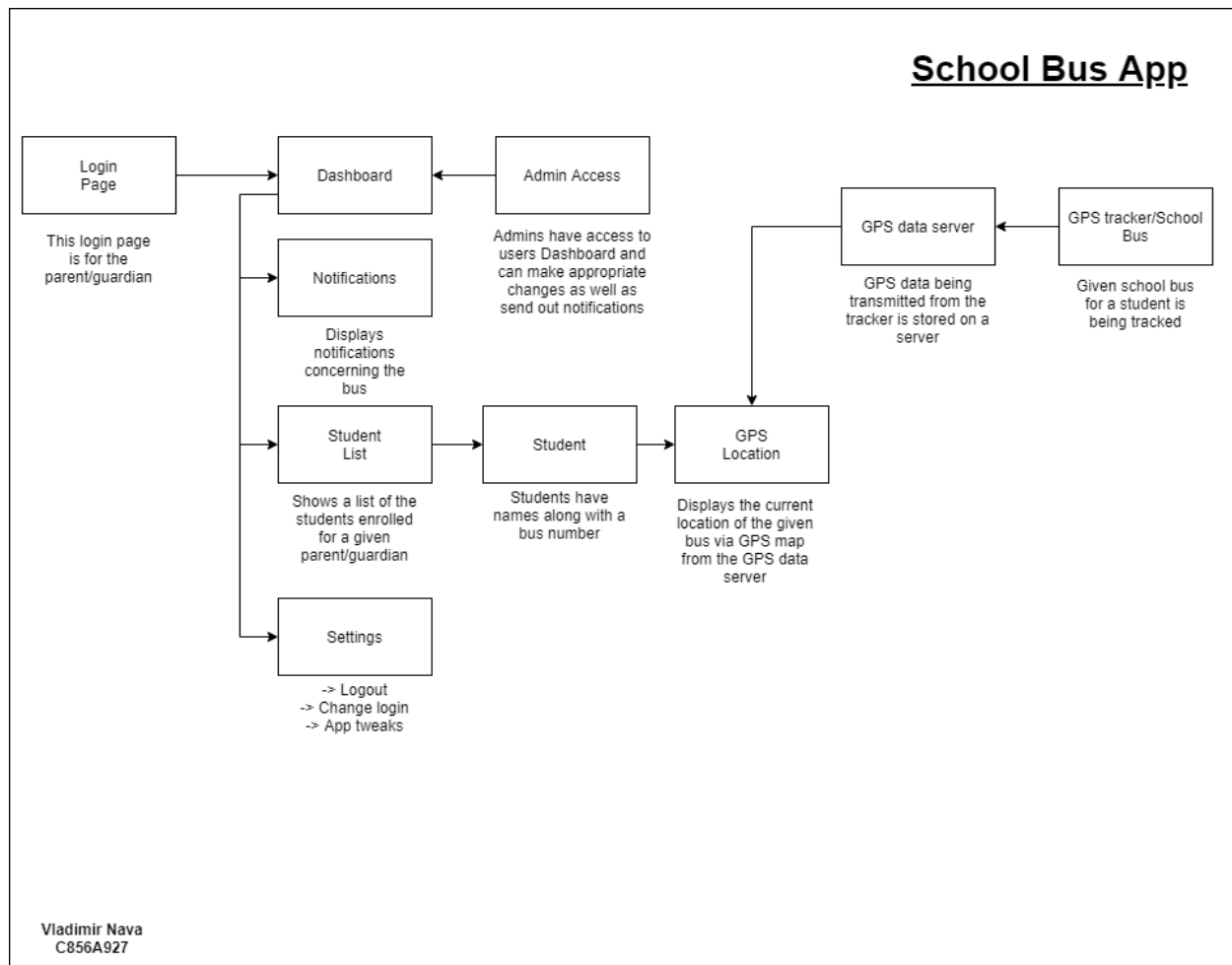
The current flow chart for the software side of the project.

The idea is to make it simple for the parent/guardian while providing important information that includes bus notifications and the location of the given bus.

The notification center should show if the bus has broken down, or if another occasion has happened concerning the student's bus.

Firebase is a service that is used by an overwhelming number of apps on the market today that helps in facilitating as well as developing apps.

For now, research is to be conducted by the team on how to implement Firebase to our project as well as any SDKs needed.



Next week I will be testing the GPS module along with the raspberry pi.

The goal is to familiarize myself with GPS data as well as plot the data via google maps.

If time permits, I will also simulate the GPS data to show functionality between the tracker and app.

Week 10/31 – 11/6

This week the team has decided on dedicating people to certain roles pertaining to the app, I am working on the backend of the application.

I have installed the GPS module onto the pi using a micro-USB to USB port.

The configuration was straight forward based on the researched conducted.

The first thing I did was remote into the raspberry pi via a laptop using ssh, vscode (Visual Studio code) and WSL (Windows subsystem for Linux).

This proved to be tricky as it was incompatible due the OS installed on the raspberry pi (old version of Raspbian). I reformatted the SD card and installed a fresh version Raspbian using the image software found on raspberry pi's website. Once properly installed, I inserted the SD card

into the pi and began to update the OS. From there working remotely was easy using a guide from raspberry pi themselves.

Working from remotely via ssh and vscode is useful as less ports are needed on my end as well as being able to visually see the file system from within vscode.

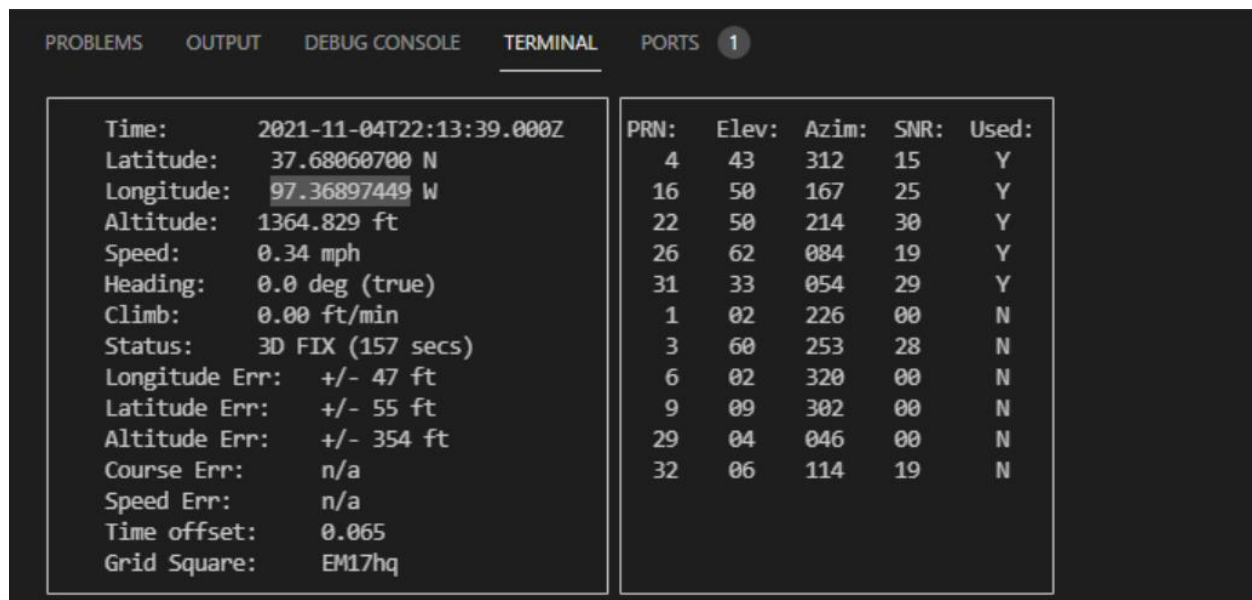
The next part was configuring the GPS module for the raspberry pi. The method that I used was to first figure out what USB port the module was connected to. This is important as specifying the USB helps in properly turning on the module for further usage.

A package was installed using the command “sudo apt-get install gpsd gpsd-clients python-gps.”

Another problem that I ran into was figuring out how to change the ownership of the files included for the gps socket. A simple search showed the command “sudo chown pi” changes the owner to the specified user.

This package is used to monitor the module as well as changing the local host for the listen stream.

Once the module was properly configured all that was left to do is to see if it was working using the command “cgps.” The following output was produced.



| | |
|--------------------------------|-----------------------------|
| Time: 2021-11-04T22:13:39.000Z | PRN: Elev: Azim: SNR: Used: |
| Latitude: 37.68060700 N | 4 43 312 15 Y |
| Longitude: 97.36897449 W | 16 50 167 25 Y |
| Altitude: 1364.829 ft | 22 50 214 30 Y |
| Speed: 0.34 mph | 26 62 084 19 Y |
| Heading: 0.0 deg (true) | 31 33 054 29 Y |
| Climb: 0.00 ft/min | 1 02 226 00 N |
| Status: 3D FIX (157 secs) | 3 60 253 28 N |
| Longitude Err: +/- 47 ft | 6 02 320 00 N |
| Latitude Err: +/- 55 ft | 9 09 302 00 N |
| Altitude Err: +/- 354 ft | 29 04 046 00 N |
| Course Err: n/a | 32 06 114 19 N |
| Speed Err: n/a | |
| Time offset: 0.065 | |
| Grid Square: EM17hq | |

The information displayed above communicates a multitude of information. From our current understanding of GPS as a team, the only relevant information to us is Latitude and Longitude as well as time.

Using the findings, the next step is to set up a Firebase project and send the data over to a real-time database.

Week 11/7 – 11/13

I created a Firebase project as well as a real-time database for conducting tests involving the GPS tracker. This involved familiarizing myself with the Firebase platform as well as learning how real-time databases work.

A real-time database can send data to users in real time. For our phone app, this is necessary for the requirements specified.

Once the Firebase project for testing was constructed, next was to create the program for the GPS tracker data.

This required me to learn more about Python as I only know the very basics of the language.

Learning Python was not too difficult and proved to be useful in that it helps in automating the process of sending the data over to the database.

The problem I have encountered so far is that the libraries needed to create the program were not installed correctly. I fixed this and the libraries now can be imported into the source code.

An important library that I have found for handling GPS data is called “pynmea2”. This library handles the NMEA GPS protocol that is a standard for GPS information. The GitHub page explains how to use the library with python as well as useful use cases for the functions given in the library.

I have learned how to send data to Firebase via another python library called “Pyrebase”.

The GitHub page again explains what important features are contained as well as how to save and send data over to Firebase. The following screenshot is from the GitHub Pyrebase page.

```
import pyrebase

config = {
    "apiKey": "apiKey",
    "authDomain": "projectId.firebaseio.com",
    "databaseURL": "https://databaseName.firebaseio.com",
    "storageBucket": "projectId.appspot.com"
}

firebase = pyrebase.initialize_app(config)
```

The above screenshot shows how to set the config for your Firebase project. Each setting configures the projects api key, the domain, URL of the database, and a storage bucket.

There are a multitude of functions that handle data as well as tokens that “adhere to security rules” (thisbejim). Security rules can be specified from within the Firebase project settings.

For next week, my goal is to get the program to run on the raspberry pi and communicate with the Firebase project.

Once this task is completed, I will relay the information to the team, and combine our efforts for the prototype presentation.

Personal Product Reflection

Our team interviewed a total of four people that includes, two parents, a high school student, and a professor. We also interviewed the school district we will be working with, Haysville school district to get a better understanding of our project requirements. The interviews were conducted individually and as a group due to time constraints and scheduling conflicts. The overall feedback given from these interviews is that the idea is welcomed and how such an idea would give parents peace of mind. For the interface side of the app, the majority have stated how ease-of-use is imperative if the app is to be successful. That is, any end user can easily identify what each functionality of the app does. From my perspective, our requirements do not need to be changed much going forward. The functionality of the app (sending notifications, tracking buses, and user friendliness) do not need to be refined as much as we previously thought. For the hardware side however, this needs to be looked at for the next semester. As the project has progressed, so to have the requirements of the hardware. The idea was to first use the school bus driver's phone as a GPS tracking device. That idea fell through as it is contingent on a lot of things, we as Engineers cannot control such as regulations. The second idea we have stuck with for this semester is to build the GPS tracker ourselves. This idea is better than our first iteration but makes for scaling the project difficult as we not only have to build the tracker, but the software installed on it as well as planning how to send the data over to a database. The third method we are considering is a combination of the two ideas so far, using a low-cost smartphone as a GPS tracker. Again, our requirements shift

which tells me that we have not clearly defined what is needed on the hardware side of the project. I do believe our understanding of the problem is wrong. We need a tracking method that adheres to our requirements set out. The requirements that I have refined our that it needs to be low-cost, low power management, scalable (can make multiple trackers), and send out data through a low-cost data plan. The key term throughout all theses requirements is that it needs to be low-key or in other words not too obtrusive for the school bus. Overall, the interviews have proved to be of great use and have been used to refine our requirements.

Work Package

For next semester, there are a multitude of items that must be completed in a timely fashion to have a final product to showcase.

- Choose a tracking method
 - There are two methods that the team is still deciding on using
 - The first is building a GPS tracker using a GPS module, raspberry pi, and a SIM card module
 - The second is to use a low-cost smartphone and repurpose it to use as a GPS tracker and choosing a cheap cellphone carrier service
 - This decision needs to be made at the beginning of the semester at the latest
- Configure GPS tracker
 - The tracker will need to have the necessary components in place so that the front-end and back-end of the project work together
 - I would like to better facilitate the data that is being collected in real time
 - That is to format the data in NMEA form
- Design the storing method
 - I will need to create a method of storing and reading the GPS data so that the front-end can communicate the necessary information
 - I will also need to write a program to manage this
- One problem that I foresee is managing multiple GPS trackers all on a single database
 - Currently we only are working with one prototype tracker that is stationary
 - I will need to learn about GPS signals as well as how to manage all the GPS data that is being transmitted
- Test the GPS tracker on the school bus
 - Currently no tests have been done on a moving object let alone a moving vehicle
 - I would like to test this as early as possible to refine the requirements of the project

References

<https://swift.org/blog/swift-on-windows/>

<https://firebase.google.com/docs/ios/setup>

https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/#//apple_ref/doc/uid/TP40006556

<https://developer.apple.com/documentation/xcode/running-your-app-in-the-simulator-or-on-a-device>

<https://www.gpsworld.com/what-exactly-is-gps-nmea-data/>

<https://searchmobilecomputing.techtarget.com/definition/Google-Firebase>

<https://www.raspberrypi.com/products/raspberry-pi-3-model-b/>

https://www.amazon.com/Microcontroller-Compatible-Sensitivity-Navigation-Positioning/dp/B07P8YMVNT/ref=sr_1_3?keywords=gps+module&qid=1636776497&qsid=143-7364663-2429625&sr=8-3&sres=B07P8YMVNT%2CB08MZ2CBP7%2CB078Y6323W%2CB084MK8BS2%2CB08636HLFG%2CB07WM1GFY8%2CB01H1R8BK0%2CB092RVFSN4%2CB0936W5B2N%2CB07PRDY6DS%2CB078Y52FGQ%2CB078Y4XZN9%2CB081YTBNLY%2CB086ZK9BT4%2CB086ZKFFY4%2CB07QPKR2FV

<https://www.raspberrypi.com/software/>

<https://www.raspberrypi.com/news/coding-on-raspberry-pi-remotely-with-visual-studio-code/>

<https://itsfoss.com/ssh-into-raspberry/>

<https://github.com/Knio/pynmea2>

<https://github.com/thisbejim/Pyrebase>

<https://tutorial.cytron.io/2020/12/09/send-data-to-firebase-using-raspberry-pi/>