Open Source

Bus Tracking System

Developed by:

Cody Johnson,

Rebecca Lawrence,

Darcy Steele

[**Overview:**](#_tcm30w45nlt2) **4**

[The Client](#_q7ghevvfrf0o) 4

[The Team](#_b34yz1uxajo1) 4

[The Task](#_ll7ic8nrntde) 4

[**Work Breakdown**](#_w3yfrix14o8s) **5**

[**Software Architecture Overview-**](#_h3p626eb9wa0) **5**

[SQL Database -](#_o99y9qt7bfae) 5

[Web application-](#_vm09nj11469p) 6

[Map components-](#_ufd35xf4ia40) 6

[Component Overview-](#_71pknt5k943h) 6

[**Detailed Design-**](#_2xmt2a6minlk) **7**

[Web View](#_b432ir9oom8i) 7

[Class Diagram-](#_5ugi33pe5z3i) 8

[Use Cases-](#_i7ten2y9j7vr) 9

[Use Case Specifications-](#_l075xs2gvpkj) 10

[User Interface Design-](#_pvey1hopbt28) 15

[Database Design-](#_lxacsn37qju6) 16

[**Review of Progress-**](#_uwhe5to8yqy3) **17**

[Revised Schedule-](#_uag226opma6l) 17

[Traceability Matrix-](#_4xu0imrlk9m4) 18

[**Testing and Validation-**](#_rqy3jwp5j1bz) **18**

[**Requirements Analysis-**](#_3s3wpl47nt0e) **19**

[Functional Requirements](#_deq05s2fg3t0) 19

[Web Interface](#_ml9zjxblmzf5) 19

[Backend](#_76djjzpndlx0) 20

[Non-Functional Requirements](#_mizz6mvma73v) 20

[**Dependencies**](#_1tdos1w5ejur) **21**

[**Deliverables**](#_3lqv14gke4y4) **21**

[**User Manual**](#_v0n0z71978se) **22**

[**Admin Guide**](#_e65lzf4jybge) **24**

[Changing the Number of Buses](#_tsg71xy1cwd9) 24

[Changing the Tracking Data of Buses](#_zfs27zrwqvtg) 24

[Changing the Data in the Tabs](#_ci8g59c72l2p) 24

[Changing the Title](#_7bu93rlsr10l) 24

[**Deployment/Handover Plan**](#_1vrups6m6abi) **24**

[**Copyright**](#_lpbyrirscr8q) **25**

[**Process**](#_7iltefv53fhi) **25**

[Developmental Methodology](#_jy5rkcwf655m) 25

[Version Control/Issue Tracking](#_d26naiyzbum9) 25

[**Visibility Plan**](#_duwd3uokkmkx) **25**

[Client](#_t18f7m9b9i0x) 25

[**Business Considerations**](#_nrgqcdna7ccb) **26**

[Copyright and Ownership](#_v301d3pmd1u0) 26

[**Risk Analysis**](#_3sex0i2vpi0k) **27**

[Team Experience](#_n08o0ld1kipm) 27

[General Delays](#_wdy8hruuljj4) 27

[Data Loss](#_ip4py02q7ohk) 27

[Change in Requirements](#_q54ttleyqn8l) 28

[Lack of Resources/Support From MCS](#_5zesaqz4l63o) 28

[**Technical Feasibility**](#_yi86ets4tznx) **28**

[Framework and Language](#_qzzrjb39ohpu) 28

[Development Environment](#_tr15augdzxag) 28

[Leaflet API](#_rx5z8iq7a4dk) 29

[**Source Code**](#_lpp33q17625z) **29**

[map.js](#_80z6uhp6ms) 29

[index.html](#_k0xrmtvgrlcx) 30

[map\_style.css](#_t69bvd6tvg6g) 32

[tab\_style.css](#_dp13ob5nyk98) 32

[tracking\_style.css](#_b3iz00p6okk5) 34

[tab\_script.js](#_mucmrk6oh3g6) 35

[map.js](#_n48c10nb1b61) 35

# Overview:

## The Client

Carolyn Dowling is an associate professor at Ball State University and a mother of two children who attend the Muncie Community School system. She reached out to us in hopes to make the Muncie Community Schools’ bus system safer for the children. She can be contacted by phone at 870-273-5812 or her email address, cbdowling73@gmail.com.

## The Team

Our team consists of three students from Ball State University. Below is a list of our names and email addresses:

* Cody Johnson, cajohnson7@bsu.edu
* Rebecca Lawrence, rjlawrence@bsu.edu
* Darcy Steele, [dtsteele@bsu.edu](mailto:dtsteele@bsu.edu)

## The Task

Our objective is a web-based application that will show authorized users of the app the location and routes of Appletree buses. The objectives required for the project as a whole to be successful will include a browser-based front end for users to view the bus status, a back-end and database designed to request, receive and store data from GPS units and send that information to the front end on user request alongside necessary map data, and hardware elements that can transmit GPS data to the back end from transmitters in the field.

The end result will be a service that allows administrators of Appletree the ability to confirm the status and expected arrival time of any given bus, allowing them to properly plan and anticipate the arrival of their children both during day-to-day bus routing as well as changes in routing caused by delays, early closures, inclement weather, etc., resulting in parents who feel more secure for knowing the location and anticipated arrival of their students.

# Work Breakdown

Cody Johnson-

* Milestone Reports
* Presentations
* Routing
* Map

Rebecca Lawrence-

* Milestone Reports
* Presentations
* GPS
* Map

Darcy Steele-

* Main point of contact with client
* Milestone Reports
* Presentations
* Map
* Website design

Completed Progress

# Software Architecture Overview-

## SQL Database -

Our project includes a database written in SQL. It holds all of the route information for each of the buses along with the user information that allows users access to the application.

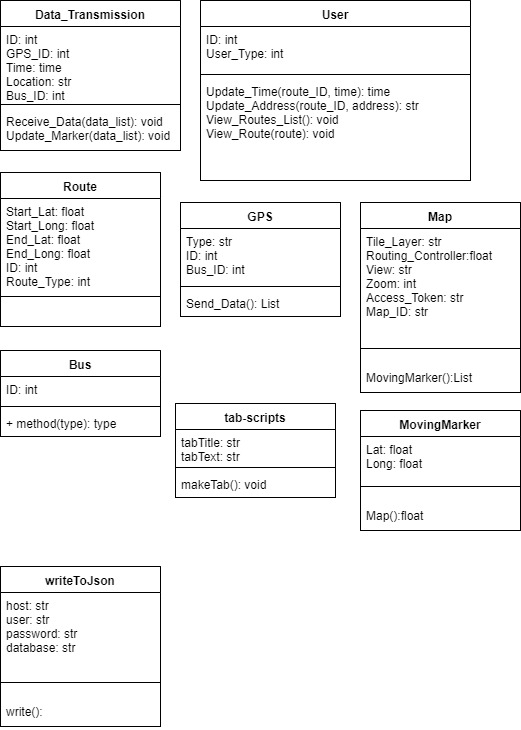
## Web application-

The web application is created with a combination of HTML, CSS, and Javascript. This allows the users to view a map of the route and view a list of all of the available buses and their estimated time of arrivals.

## Map components-

The map that is displayed on the application is made with Leaflet, Mapbox, Openstreet. We use the Leaflet framework and plugins which use MapBox and OpenStreet.

## Component Overview-

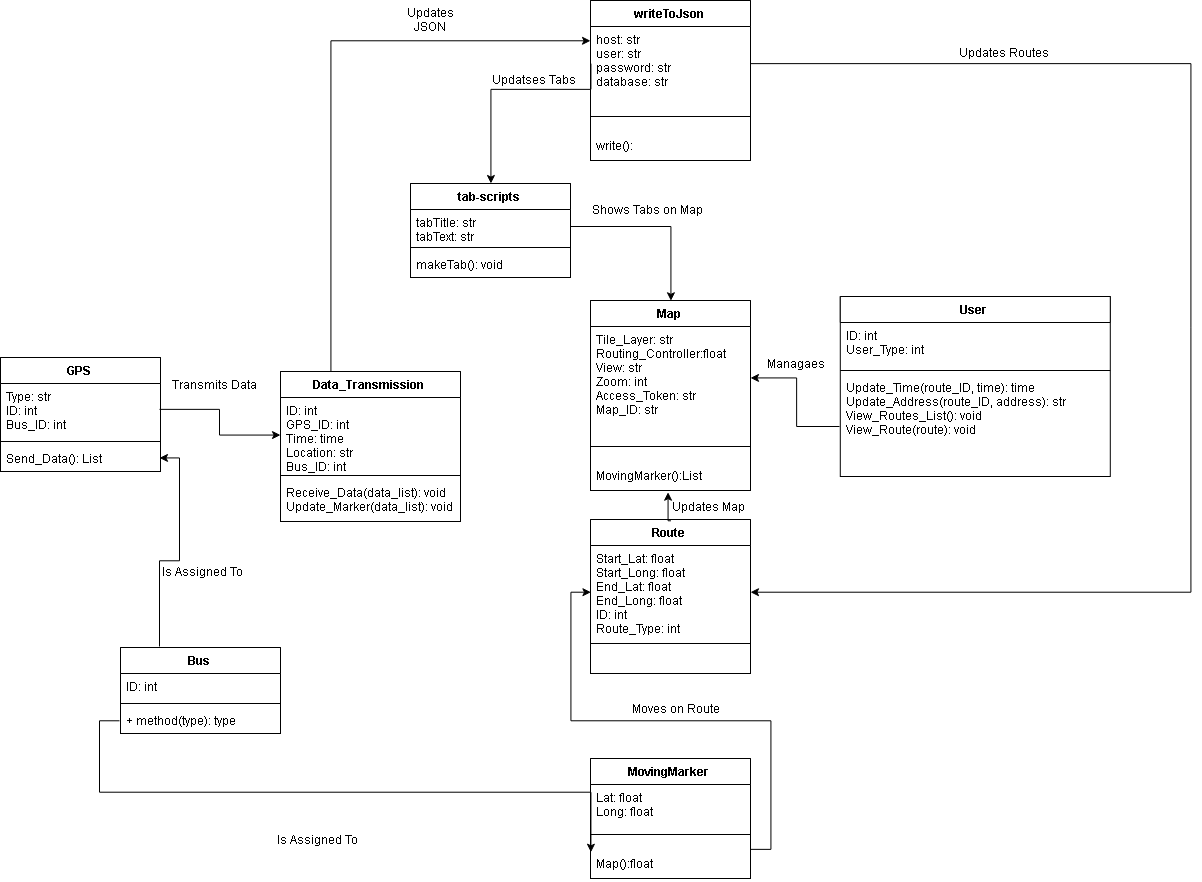


# Detailed Design-

## Web View-

* index.html - The main file for the Web View, it is a file that contains all of the HTML elements for the Web View.
* map.js - The Javascript file that controls how the map on the Web View functions. It connects Leaflet and the Plugins for Leaflet to the Web View.
* tab\_script.js - The Javascript file that controls how the tabs on the Web View function. It adds tabs to index.ejs so that they show in the Web View. Each tab contains the details for a bus, such as the bus name and estimated arrival time to AppleTree.
* map\_style.css - The css file that controls how the map on the Web View appears.
* tab\_style.css - The css file that controls how the tabs on the Web View appear.
* tracker\_style.css - The css file that controls how the entire Web View appears.
* Leaflet - A framework that is being used to control the map for the Web View.
* Leaflet-Routing-Machine - A plugin for Leaflet. It adds routing functionality to the map on the Web View.
* Leaflet.Tracking - A plugin for Leaflet. It adds a marker on the map that moves to a location from an url.

## Class Diagram-



## Use Cases-

#### 

## Use Case Specifications-

Case 1:

Use Case name: ViewEta

Actors: users

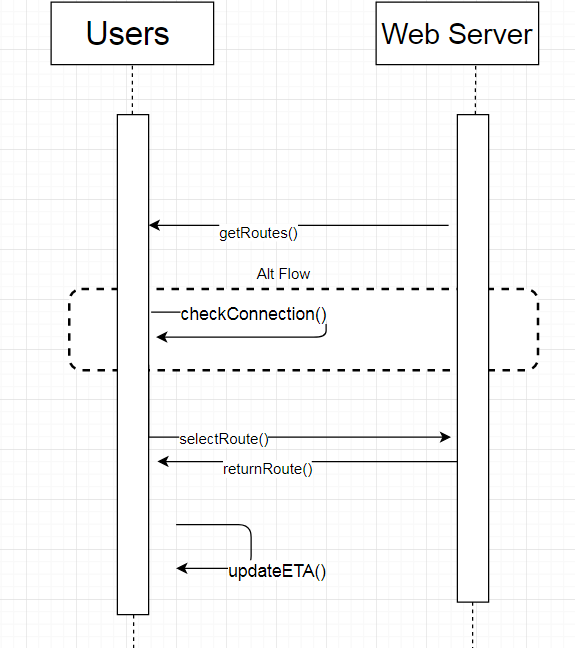
Overview: As a user I want to be able to know when my bus will arrive.

Typical Flow of Events:

1. user selects route from left hand side of screen
   1. user selects desired bus
2. Selected route (path of route, current position of bus) appears on map display
3. Information is shown regarding the scheduled time the bus is supposed to arrive .
   1. Information is also shown based on the bus’s current position to determine if bus may be early or late

Alternate Flow:

1a. User is not connected to the internet.



Case 2:

Use Case name: ViewRoutes

Actors: users

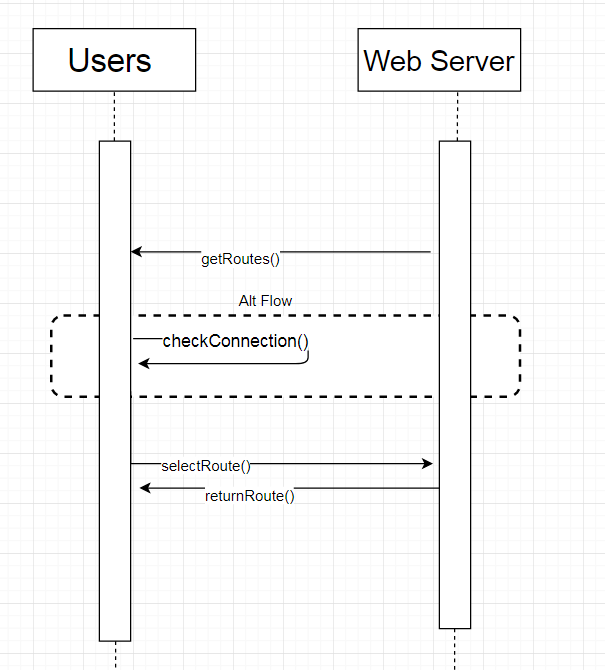
Overview: As a user I want to be able to see the route my bus will take.

Typical Flow of Events:

1. user selects route from left hand side of screen
   1. user selects desired bus
2. Selected route (path of route, current position of bus) appears on map display

Alternate Flow:

1a. User is not connected to the internet.



Case 3:

Use Case name: ViewBusPosition

Actors: users

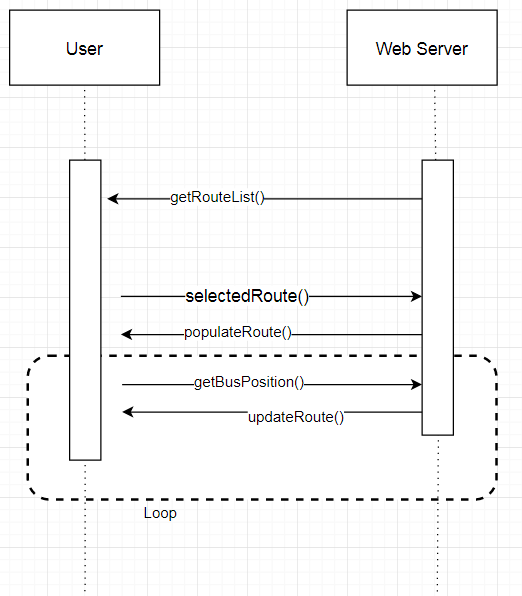
Overview: As a user I want to be able to see the buses location in real time.

Typical Flow of Events:

1. user selects route from left hand side of screen
   1. user selects desired bus
2. Selected route (path of route, stops, current position of bus) appears on map display
3. Information is shown regarding the scheduled time the bus is supposed to arrive
   1. Information is also shown based on the bus’s current position

Alternate Flow:

1a. User is not connected to the internet.



Case 4:

Use Case name: UpdateBusRoutes

Actors: Administrators

Overview: As an administrator I want to update bus routes when they change.

Typical Flow of Events:

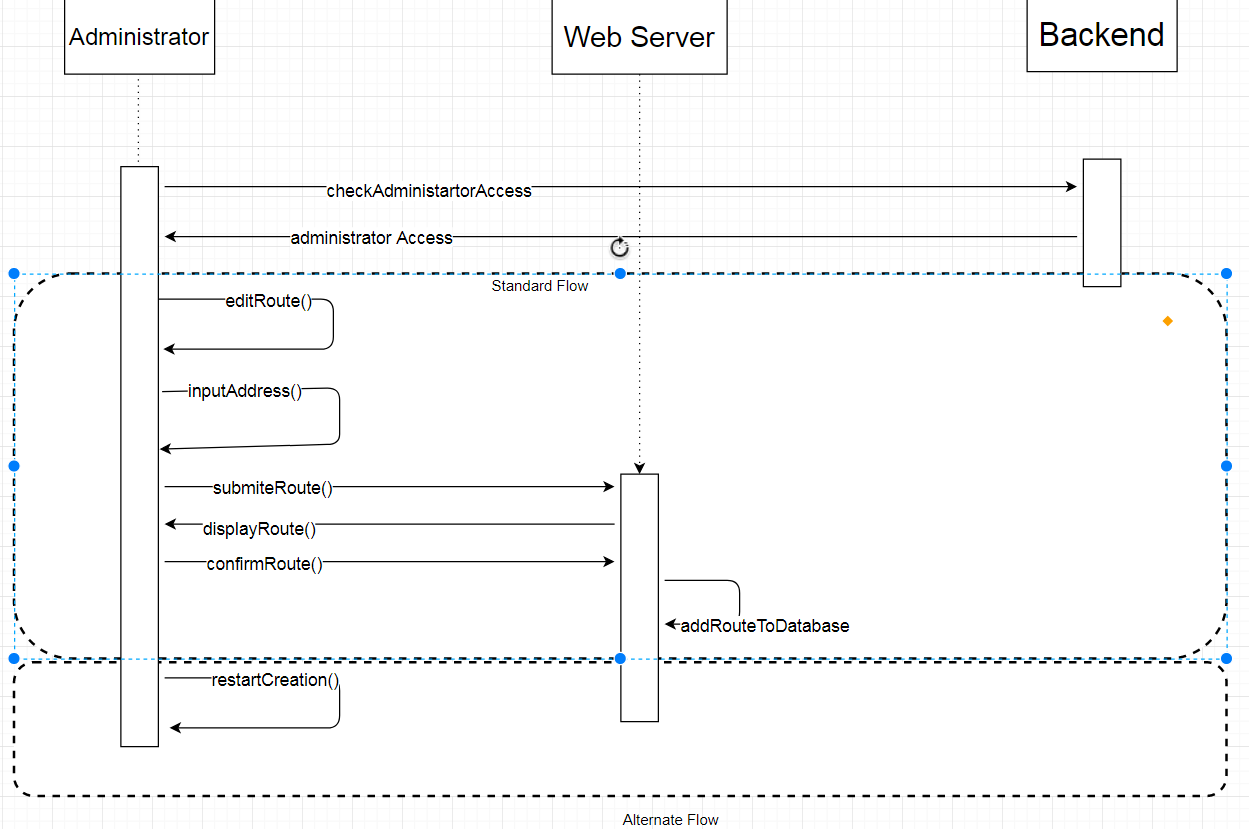
1. Administrator will click on the edit button above the map
2. They will enter the address for the starting location
   1. Press next when finished entering the address
3. Enter the end address
   1. Press next to enter to finalize route
4. The route will display on the map and will show up in the list\

Alternate Flow:

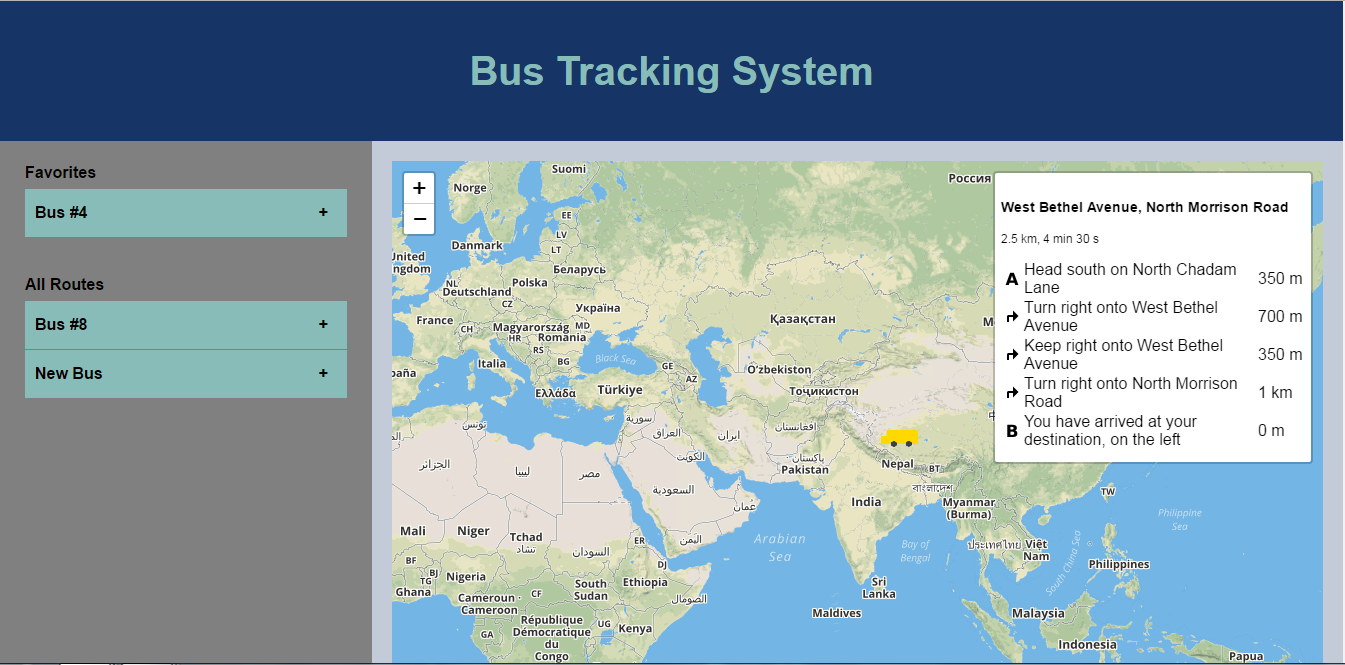
1a. User has not yet entered credentials to access content.

2a. The address entered is invalid.

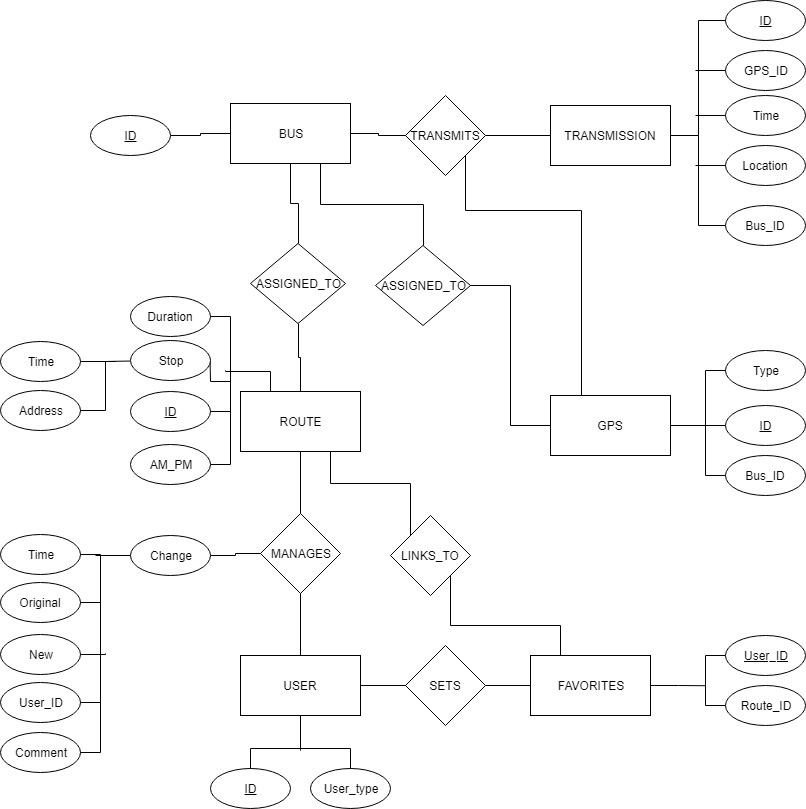
4a. The route was incorrect and admin must restart the creation.



## User Interface Design-



## Database Design-

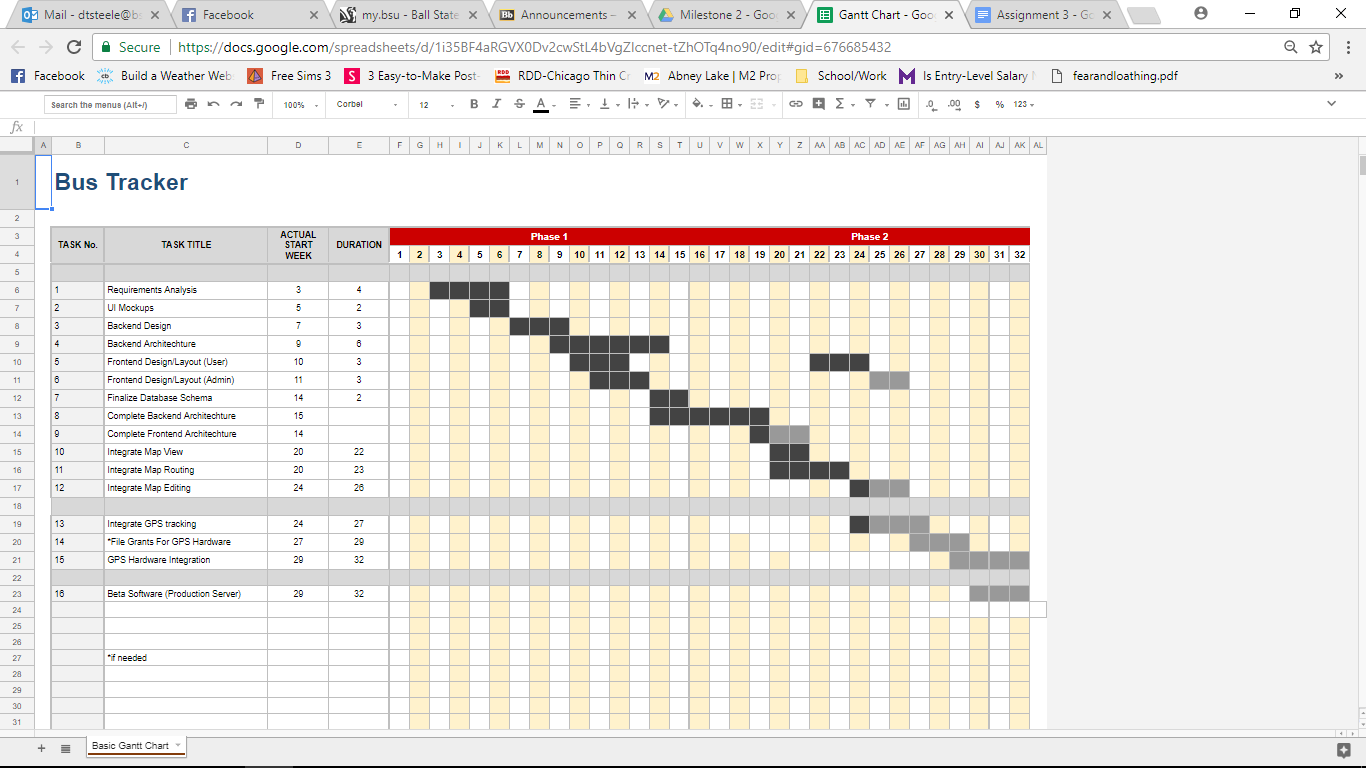


# 

# Review of Progress-

* Front end application is finished with a basis for the connection to the GPS
* Set up a test tracker to hand over to the Client.

## Revised Schedule-



# 

## Traceability Matrix-

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Traceability Matrix** | | | | | | | | | | | | | | | | | | |
| **Case Number** | **Requirement Number** | | | | | | | | | | | | | | | | | |
| **1F** | | **2F** | | | | | **3F** | | **4F** | | | | **5F** | | | | |
|  | **a** | **b** | **a** | **b** | **c** | **d** | **e** | **a** | **b** | **a** | **b** | **c** | **d** | **a** | **b** | **c** | **d** | **e** |
| **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

(Figure 2. Traceability Matrix)

# Testing and Validation-

Manual Testing- Due to constantly changing requirements and lack of understanding our group did most of its testing manually. We used a test document with example gps and map location data.

Mobile Testing- Because our project is meant to be able to be run from both web and mobile we have tested our projet on several phones in order to make sure that it functions properly on them.

User Testing- Overall our user testing lead us to changing how our project looked. We started out with a very bright and colorful background, but users found that to be distracting from the overall use. After changing the color scheme of the website subsequent users had less criticism because of the specificity of the project. A few key problems that were pointed out was that currently the buses are not interactable and that there is an occasional uneven zoom out problem.

# Requirements Analysis-

## Functional Requirements

### Web Interface

1F. User Access (Required)

* 1. Only people with a school account privileges should be able to access the application using their existing school account.

2F. Routes (Required)

1. The app, after login, will show the user a list of buses.
2. Each route will show the list of stops for that bus route on the map along with its scheduled bus arrival time.
   1. After being selected, the map should show the user the current position of the bus along its route.
   2. Each bus stop on the map should be selectable to show the user the estimated arrival time.

3F. Routes (Optional)

1. After selecting a route, the app will query the database to find the current position of the bus.

4F. Administrative Access (Required)

1. The web app, upon recognizing an admin login, should present additional options to add and edit routes.
2. Selecting to add a route will give the user the ability to pair a GPS unit with a new route in the database, then allow the admin user to continue editing the route.
3. When editing routes, the user will be able to reposition waypoints or add new waypoints, then order them to represent the route a bus takes.
4. The web app will also have a feature for admin users to temporarily assign a bus to a different GPS tracker, different route or both.

### Backend

5F. GPS Handler

* 1. The backend must feature a service to send/receive SMS/MMS messages to GPS units in the field.
  2. The GPS handler must be able to receive a request to message a GPS unit in the field, consult the database to determine the format the message must take, send the message to the receiving GPS unit and confirm that the operation is completed.
  3. Upon receiving a message from a GPS unit in the field, the GPS handler must consult the database to determine the expected format of the message, parse the message into the data format required by the database and then insert that data into the database.
  4. The GPS handler must be able to confirm the total duration of any send/receive operation for the purposes of troubleshooting GPS units in the field.
  5. The GPS handler must be able to intelligently handle errors in sending/receiving messages - if units stop responding, if units send multiple replies to a single request, etc.

## Non-Functional Requirements

1N. The web app must run on Android and iOS devices as well as in common desktop browsers at common mobile and desktop resolutions.

2N. The web app must be built on the Node.js framework.

3N. The web app’s map view must use the Leaflet API to insure up-to-date map information.

4N. The web app must have either its own login service or, ideally, tie in to the MCS’s login system.

5N. The web app must be easy to use by untrained users, i.e. parents of students and administrators.

* 1. The GUI should be designed in a fashion that presents users with an unambiguous path to follow to their desired outcome.
  2. Administrators must be able to edit and change bus and route data easily using the provided GUI.

6N. The backend must properly transmit/receive GPS data in multiple formats for use by the front end.

1. The backend must also feature a mock GPS transmitter to feed data into the database/front end for troubleshooting/testing.

7N. Code for the frontend and backend must be extensively documented for future maintainers to use.

8N. The app must make only necessary calls to the backend, to minimize bandwidth usage and insure that the app is efficient and responsive.

9N. The app must meet common security standards as well as whatever further standards MCS requires to meet any legal or ethical liabilities and obligations they may have.

10N. The app must be reliable with reasonable steps taken to insure that the app’s uptime is maximized.

1. The app and its associated backend must be configured to restart without administrator input in the event of server restart, crash or other failure.

# 

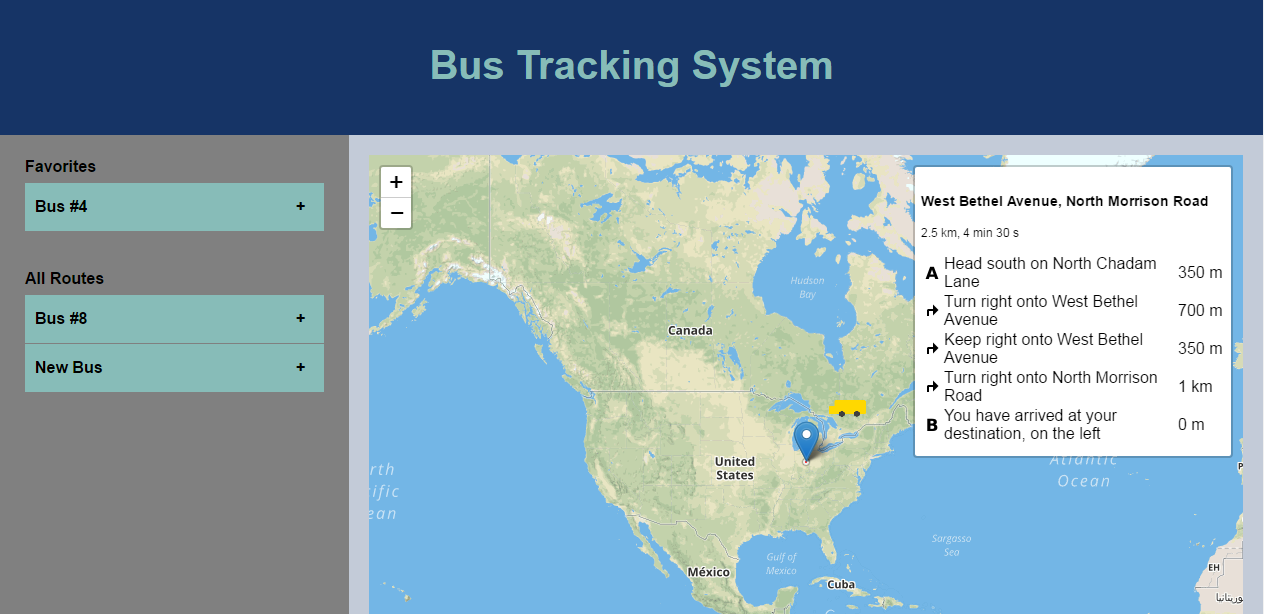
# Dependencies

* Leaflet
* Leaflet-Routing-Machine
* Leaflet Realtime
* OpenStreetMap
* Mapbox

# Deliverables

* Bus Tracking Web Application
  + Should everything be okay with the school system, we will deliver a web application that tracks the Appletree bus fleet with given hardware..
* General GPS Tracking Web Application/But Route Web Application
  + Should Appletree not allow us the necessary access to the buses or access to a server to host the application on, we are planning on making a similar web application that allows us to track and manage GPS transmitters.
  + In addition, to best meet the intent of our client’s request, we will deliver a similar web app that presents the routes and expected arrival times of buses, using publically available route data from MCS.

# User Manual



Thank you for using our Open Source Bus Tracker. This guide will help you understand how to use the application if you are unfamiliar with the format. This application is designed to provide a low cost option for tracking busses or other vehicles that people or organizations may be unable to afford the more expensive options for. This tracker was created by Cody Johnson, Rebecca Lawrence, and Darcy Steele.

|  |  |
| --- | --- |
|  | This is the title of your bus tracking application. If you are an administrator and would like to learn about editing this consult the Admin Guide. |

|  |  |
| --- | --- |
|  | This is the map view. This allows you to see where the bus you are tracking is currently located and provides directions if you need to know the route the bus is taking. It also allows you to zoom in and out. |

|  |  |
| --- | --- |
|  | This is the bus marker. It shows you the position of the bus. If you are an admin and want to know more about the marker and changing it to a different image, refer to the Admin Guide. |

|  |  |
| --- | --- |
|  | This is the sidebar that houses all of the information you will need to know about the busses. It shows the bus name and if you click the plus sign that is pictured on each individual tab in the image to the left, you will be able to see the bus’ estimated time of arrival. There is also the option of adding favorite busses to a list at the top of the sidebar so they are always easily accessible. If you are an administrator and would like to know more about how to change the bus list or remove features, refer to our Admin Guide. |

# Admin Guide

### Changing the Number of Buses

Currently to change the number of buses you must go into the map.js file and copy the "var = realtime" code and add it as many times as the buses that you need.

### Changing the Tracking Data of Buses

To change the data that is connected to a bus, find the "url" line in the corresponding "var = realtime" and change it to the file path the contains the tracking data.

### Changing the Data in the Tabs

Currently to add more tabs run the makeTab() function for the amount of buses. To change the data, there are two parameters in the function, a "name" and a "data". The name controls the data on the tab when closed, the data controls the data in the closed tab.

Example: makeTab("Bus 1", "Bus Destination")

### Changing the Title

To change the title, find the div class "page-title" and change the "h1" tag to the prefered title

# Deployment/Handover Plan

Website stored on a temporary server.

We will handover the source code and documentation to Carolyn.

Appletree can then choose to keep the hosting server or transfer it to one that they own.

# Copyright

As talked about prior our project was built on the concept of being open source from the start. Anyone who wishes to use or update the project is free to do so.

# Process

## Developmental Methodology

Our team has decided to use one of the agile software development methodologies, specifically Extreme Programming (XP). XP provides us with the ability to create a very collaborative team environment focused on short, incremental release cycles allowing us to react to client requests or new concepts we learn as we work. Test driven development will be used allowing us to insure functionality of the codebase from start to finish. Keeping our client in the loop throughout the project will insure our work is focused on the client’s wishes, factoring in her input and insuring that she’s satisfied with the output at every point.

## Version Control/Issue Tracking

We will be using Git via Github for version control. Our repository will be maintained by the group as a whole and we will be constantly updating our project as we develop new iterations. When someone completes a task that we have scheduled and pushes it to the Git repository, they will be asked to consult the group before merging their code into the master branch alongside insuring that all necessary unit tests pass. Only when the group comes to a consensus that the task is complete can the task be marked as completed on the Gantt chart. GitHub has a built-in issue tracker, Issues, which we will use to keep any noted issues with the codebase alongside the source code itself.

# 

# Visibility Plan

## Client

Darcy Steele has been designated to be our main point of communication with our client, Carolyn Dowling. Darcy has agreed to CC all team members on communication; in addition, it has been decided that other groups members can contact the client (while still CCing all team members) if there is an immediate need or if Darcy is for some reason unable to. There is currently no set recurring meeting time with the client, although both team and client have acknowledged that recurring meetings should be established once work begins on the process of communicating and negotiating with Muncie Community Schools, developing the product, etc.

Carolyn Dowling has also expressed intent to speak with other parents and school officials to build support for the project, and has experience with filing for and obtaining educational grants that may be needed to fund GPS trackers for testing and/or deployment — MCS is currently dealing with financial shortfalls, and it seems very unlikely that they will be willing or able to fund hardware needs.

Assuming Muncie Community Schools is amenable to our project, it is likely that the team will have some sort of liaison or liaisons with the school for administrative or technical needs. A similar structure to communication will be taken with these liaisons as there is with client communication, possibly being the domain of a team member other than Darcy to avoid overly burdening her with communication responsibilities.

# Business Considerations

## Copyright and Ownership

The team has indicated interest in open-sourcing the project on its completion. The nature of the project may be of interest for other schools or similar entities; a free software system paired with cheap, easily installable GPS units may be of interest to schools with little cash but a willingness to adopt and refine the product. In addition, open-source software means that as MCS’s needs change, they will either be able to use new and updated features added by the open-source community or possibly contact another capstone group for assistance in updating and maintaining the software.

It is conceivable, albeit unlikely, that MCS may disagree with the software being open-sourced for one reason or another. In the event that this occurs, the team will negotiate with MCS to find a suitable way of transferring ownership and rights to the software to MCS, with limited rights for the team members to use the source code for portfolios and other non-infringing uses of the source code or other elements of the finished project.

# 

# Risk Analysis

There are several potential risks that can cause levels of disruption ranging from minor to serious. As a team, we are confident that we have identified the likely risks that may arise and have plans to manage or mitigate any disruptions that arise.

## Team Experience

The team as a whole does not have significant experience developing this kind of project. Specifically, the team has not done any significant work with web development or management/utilization of GPS data. However, all the tools we have selected to use are established and well-documented, with a wide variety of resources available to us. At the time scale this project is established on, learning material as we go should not cause significant difficulty; in addition, as the team will be learning much of this together, we will be able to collaborate and identify problems together as they arise, rather than relying on one particular team member to handle the work.

## General Delays

Unexpected delays from a number of different sources can hamper any large project. While the end result is well-conceived as laid out in this document, at this point the client does not have any particular deadlines for iterations or deliverables other than the eventual completion at the end of the capstone course. The project as conceived does not currently have any tasks or subtasks projected to cause serious delay, and the current schedule as established by the team is purposefully over-estimated for “worst-case” scenarios.

## Data Loss

Loss of source code or other assets would be a serious setback to timely progress on this project. Proper version control system usage combined with storing design documents on Google Drive for reference by group members all but completely mitigates this risk. As the data for the project is stored with GitHub Inc. and Google Inc., there is no anticipated scenario in which total data loss could occur.

## Change in Requirements

It is possible that the client or MCS may develop differing ideas about the desired end result of the project as time progresses. Frequent and detailed communication with the client as well as any MCS representatives assigned to assist or oversee us should catch these divergences fairly early and enable us to either adapt to them or modify the expectations of the client or other parties.

## Lack of Resources/Support From MCS

Our client desires a project that interacts with the Muncie Community School bus fleet, but is not actually an administrator or policy-maker with the school district. While our desired goal is a functioning application serving reliable data to parents of MCS students, it is possible that MCS will not be willing or able to give us all the resources necessary to complete the project as expected. The client has discussed this with us and is willing to reduce the scope of the project as necessary. At the very minimum, the project will still be a functioning, tested webapp with the ability to track vehicles or other GPS-enabled assets. Functionality can be insured by mocking bus routes and movement, making the project a proof-of-concept that can be used to attempt to convince MCS or another organization to use the application or enable further development/testing.

# 

# Technical Feasibility

## Framework and Language

Use of the popular JavaScript-based Node.js framework will assist in producing a reliable product that is easy for potential future maintainers to change and update. Node.js will provide a unified framework for working with the frontend and backend and has an abundance of available documentation.

## Development Environment

The team has elected to use Brackets, an editor designed for HTML and Javascript. We chose Brackets for its preview option, allowing us to visualize the application as it is being written.

## Leaflet API

The Leaflet API is well-documented and open source app for displaying map and location data. The interface it presents is familiar to any user who has used Google Maps or any other app that uses it and should insure that users have a smooth experience learning to use the app and its features.

# Source Code

## map.js

var map = L.map('mapid').setView([40.2224765, -85.4329497], 12);

L.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.png?access\_token={accessToken}', {

attribution: 'Map data &copy; <a href="http://openstreetmap.org">OpenStreetMap</a> contributors, <a href="http://creativecommons.org/licenses/by-sa/2.0/">CC-BY-SA</a>, Imagery © <a href="http://mapbox.com">Mapbox</a>',

maxZoom: 15,

id: 'mapbox.streets',

accessToken: 'pk.eyJ1IjoiZHN0ZWVsZTIwMTQiLCJhIjoiY2phdHY3ejRtMXp2YjJ4bWlpdXRidGZpNSJ9.gdQdIvN\_1N0mlrCFnwjLCw',

}).addTo(map);

L.Routing.control({

waypoints: [

L.latLng(40.2224765, -85.4329497),

L.latLng(40.2324765, -85.4439497)

],

router: L.Routing.mapbox('pk.eyJ1IjoiZHN0ZWVsZTIwMTQiLCJhIjoiY2phdHY3ejRtMXp2YjJ4bWlpdXRidGZpNSJ9.gdQdIvN\_1N0mlrCFnwjLCw'),

}).addTo(map);

var geojsonMarkerOptions = {

radius: 18,

fillColor: "#ff7800",

color: "#000",

weight: 1,

opacity: 1,

fillOpacity: 0.8

};

var realtime = L.realtime({

url: './public/testData.json',

crossOrigin: true,

type: 'json'

}, {

interval: 3 \* 1000,

pointToLayer: function (feature, latlng) {

return L.marker(latlng, {

'icon': L.icon({

iconUrl: "./public/res/transparentBus.png",

iconSize: [120, 80], // size of the icon

iconAnchor: [22, 94],

})

});

}

}).addTo(map);

realtime.on('update', function(e) {

var coordPart = function(v, dirs) {

return dirs.charAt(v >= 0 ? 0 : 1)

(Math.round(Math.abs(v) \* 100) / 100).toString();

};

map.fitBounds(realtime.getBounds(), {maxZoom: 3});

});

## index.html

<DOCTYPE html PUBLIC "en">

<html>

<head>

<title>Bus Tracker</title>

<link rel="stylesheet" href="public/css/tracker\_style.css">

<link rel="stylesheet" href="public/css/map\_style.css">

<link rel="stylesheet" href="public/css/tab\_style.css">

<script src="public/scripts/tab\_script.js"></script>

<link rel="stylesheet" href="https://unpkg.com/leaflet@1.2.0/dist/leaflet.css" />

<script src="https://unpkg.com/leaflet@1.2.0/dist/leaflet.js"></script>

<link rel="stylesheet" href="https://unpkg.com/leaflet-routing-machine@latest/dist/leaflet-routing-machine.css" />

<script src="https://unpkg.com/leaflet-routing-machine@latest/dist/leaflet-routing-machine.js"></script>

<script type="text/javascript" src="http://www.liedman.net/leaflet-realtime/dist/leaflet-realtime.js"></script>

</head>

<body>

<div class="page-title">

<h1>Bus Tracking System</h1>

</div>

<div class="page-content">

<div class="side-bar">

<div class="tab-title">

<p>All Routes</p>

</div>

</div>

<script type="text/javascript">

makeTab("Bus 1","Appletree to Muncie Schools");

makeTab("Bus 2","Muncie Schools to Appletree");

</script>

<div class="main-content">

<div id="mapid"></div>

<script type="text/javascript" src="public/scripts/map.js"></script>

</div>

</div>

</body>

</html>

## map\_style.css

#mapid{

width: 100%;

height: 100%;

}

## tab\_style.css

.tab{

position: relative;

margin-bottom: 1px;

width: 100%;

color: black;

overflow: hidden;

}

input{

position: absolute;

opacity: 0;

z-index: -1;

}

label{

position: relative;

display: block;

padding: 0,0,0,1em;

background: #87bcb8;

font-weight: bold;

line-height: 3;

cursor: pointer;

padding-left: 10px;

}

.tab-title{

position: relative;

background: gray;

font-weight: bold;

line-height: 1px;

padding: 15px;

padding-left: 25px;

padding-right: 25px ;

padding-bottom: 15px;

}

.tab-content {

max-height: 0;

overflow: hidden;

background: #b8baba;

-webkit-transition: max-height .35s;

-o-transition: max-height .35s;

transition: max-height .35s;

}

.tab-content p {

margin-left: 2em;

}

input:checked ~ .tab-content {

max-height: 10em;

}

label::after {

position: absolute;

right: 0;

top: 0;

display: block;

width: 3em;

height: 3em;

line-height: 3;

text-align: center;

-webkit-transition: all .35s;

-o-transition: all .35s;

transition: all .35s;

}

input[type=checkbox] + label::after {

content: "+";

}

input[type=radio] + label::after {

content: "\25BC";

}

input[type=checkbox]:checked + label::after {

transform: rotate(315deg);

}

input[type=radio]:checked + label::after {

transform: rotateX(180deg);

}

## tracking\_style.css

body{

font-family: Arial,Times,serif;

margin: 0px;

background-color: #c3cbd8;

padding: 0px;

border: 0px;

}

h1{

text-align: center;

color: #87bcb8;

font-size: 40px;

font-family: Arial, sans-serif;

font: Arial;

width: 100%;

height: 140px;

vertical-align: middle;

line-height: 140px;

background-color: #163466;

margin: 0;

}

.side-bar{

width: 40%;

height: 105%;

background: gray;

}

.page-content{

display: inline-flex;

height: 100%;

width: 100%;

padding: 0px;

border: 0px;

margin: 0px;

}

.main-content{

padding: 20px;

height: auto;

width: 100%;

}

## tab\_script.js

function makeTab(name,data){

var tabTitle = document.getElementsByClassName("tab-title");

var tab = document.createElement("div");

tab.className = "tab";

var input = document.createElement("input");

input.setAttribute("id","tab-new");

input.setAttribute("type","checkbox");

input.setAttribute("name","tabs");

var tabLabel = document.createElement("label");

tabLabel.setAttribute("for","tab-new");

var node = document.createTextNode(name);

tabLabel.appendChild(node);

var tabContent = document.createElement("div");

tabContent.className = "tab-content";

var tabPara = document.createElement("p");

var node = document.createTextNode(data);

tabPara.appendChild(node);

tabContent.appendChild(tabPara);

tab.appendChild(input);

tab.appendChild(tabLabel);

tab.appendChild(tabContent);

tabTitle[0].appendChild(tab);

}

## map.js

var map = L.map('mapid').setView([40.2224765, -85.4329497], 12);

L.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.png?access\_token={accessToken}', {

attribution: 'Map data &copy; <a href="http://openstreetmap.org">OpenStreetMap</a> contributors, <a href="http://creativecommons.org/licenses/by-sa/2.0/">CC-BY-SA</a>, Imagery © <a href="http://mapbox.com">Mapbox</a>',

maxZoom: 15,

id: 'mapbox.streets',

accessToken: 'pk.eyJ1IjoiZHN0ZWVsZTIwMTQiLCJhIjoiY2phdHY3ejRtMXp2YjJ4bWlpdXRidGZpNSJ9.gdQdIvN\_1N0mlrCFnwjLCw',

}).addTo(map);

L.Routing.control({

waypoints: [

L.latLng(40.2224765, -85.4329497),

L.latLng(40.2324765, -85.4439497)

],

router: L.Routing.mapbox('pk.eyJ1IjoiZHN0ZWVsZTIwMTQiLCJhIjoiY2phdHY3ejRtMXp2YjJ4bWlpdXRidGZpNSJ9.gdQdIvN\_1N0mlrCFnwjLCw'),

}).addTo(map);

var geojsonMarkerOptions = {

radius: 18,

fillColor: "#ff7800",

color: "#000",

weight: 1,

opacity: 1,

fillOpacity: 0.8

};

var realtime = L.realtime({

url: './public/testData.json',

crossOrigin: true,

type: 'json'

}, {

interval: 3 \* 1000,

pointToLayer: function (feature, latlng) {

return L.marker(latlng, {

'icon': L.icon({

iconUrl: "./public/res/transparentBus.png",

iconSize: [120, 80], // size of the icon

iconAnchor: [60, 40],

})

});

}

}).addTo(map);

var realtime = L.realtime({

url: 'https://wanderdrone.appspot.com',

crossOrigin: true,

type: 'json'

}, {

interval: 3 \* 1000,

pointToLayer: function (feature, latlng) {

return L.marker(latlng, {

'icon': L.icon({

iconUrl: "./public/res/transparentBus.png",

iconSize: [120, 80], // size of the icon

iconAnchor: [60, 40],

})

});

}

}).addTo(map);