Michael G Menard, Brandon N Collins, Jacob D Jones

Keep Kids alive map system admisistration manual

Keep Kids Alive Map System Administration Manual

| Table of Contents |

1. General Information - 2
   1. System Overview - 2
   2. Organization of the manual – 2-3
2. The Submittal Page - 3

2.1 The Submission Page - 3

2.2 The Buttons - 4

3.0 The Map Page - 4

3.1 The Map API – 5-6

3.2 Home.php – 6-7

4.0 The Contact Page - 7

4.1 The Contact Page and the Form – 7-8

5.0 The About page- 8

6.0 The Table Page - 8

6.1 The Construction of the Table – 8

7.0 The Navigation Menu – 8

7.1 Adding Additional Links to the Navigation Bar – 8-9

8.0 The SQL Table and How it is Structured – 9

8.1 Basic Structure - 9

8.2 Removing Points From the Table – 9-10

1. General Information

This section of the Manual will provide general information regarding the system, including its uses and how to use the interactive portions of the system.

* 1. System Overview

The Keep Kids Alive Map is an application which allows the collection of information regarding minor traffic infractions. It provides a simple, one button press submission method for several major traffic infractions, as well as an interactive map created with the purpose of visualizing the data. It also provides a table of submitted data for more in-depth analysis. All data collected is saved to a database. The app was designed with mobile devices in mind, and is designed for dynamic growth in most areas. This means that extra points won’t have to be manually made on the map, and the need to include extra data manually is completely obsolete. The storing and acquisition of data is entirely automated and is designed to support any number of data points. Some parts of its functionality may not work properly on desktop systems.

* 1. Organization of the Manual

This manual consists of several different sections: The submittal page, the Map Page (home page), the Contact Page, the About Page, and the Table page. Each section will provide descriptions of the inner workings of each of the listed pages, including their content, how to add additional functionality to them, what parts of them are dynamic, and what files they reference if they need to acquire data.

The submittal page is how data is submitted to the database, and uses the Interactive\_page.php document to submit the data according to the button pressed on this page.

The Map page makes use of the home.php page to acquire the data from the table on the mysql database and transforms that data into points on the map.

The About Page describes what Keep Kids Alive Drive 25 does.

The Contact page uses a contact form and the contactus.php file to send an email to the proprietor of the app containing messages from users.

The Table Pages uses the same home.php file as the Map Page to generate an analog of the table available on the mysql server for general public viewing.

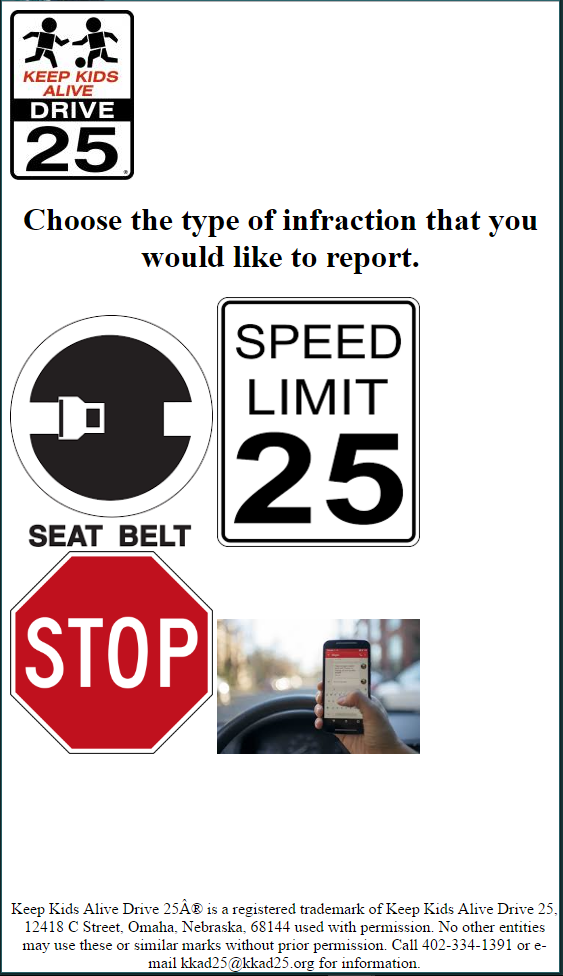
The SQL Table section describes the workings of the SQL table, the types of data each column is comprised of, and a couple of warnings regarding the table and the dynamic scripts available on other pages.

The Navigation menu section details the menu used for navigation and how to add pages to the list.

1. The Submittal page

This section will detail the script behind the submittal page, how to add more buttons should more infractions want to be added, and how it sends the data to the mysql database using the interactive\_page.php document.

* 1. The Submission Page

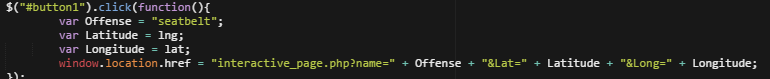


Button for submitting infractions related to unbuckled seatbelts

Button for submitting infractions related to distracted driving

Button for submitting infractions related to speeding

Button for submitting infractions related to rolling stops.

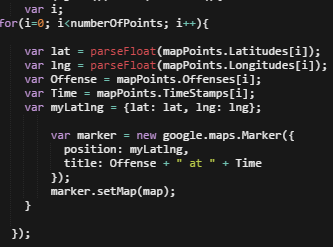
* 1. The Buttons

The buttons outlined on the page use a simple script to send the data from the interactive\_page.html file to the interactive\_page.php file. Upon being clicked, the button uses lat and lng variables containing the latitude and longitude location of the user, as well as an offense variable containing the ascribed offense for that button, and puts them into the header of the interactive\_page.php file. Using $GET methods, the file pulls that data down and then connects to the php server using mysqli\_connect(). Then using the mysqli\_query, it queries the mysql server to put the data received from the button click into the table available on the server. To add another button, copy the displayed script and change the offense variable to the kind of offense you wish to log. Then add another button in the html that references the ID of this button. Since the SQL table just logs strings for the offenses, no other changes should need to be made for the button to function properly.

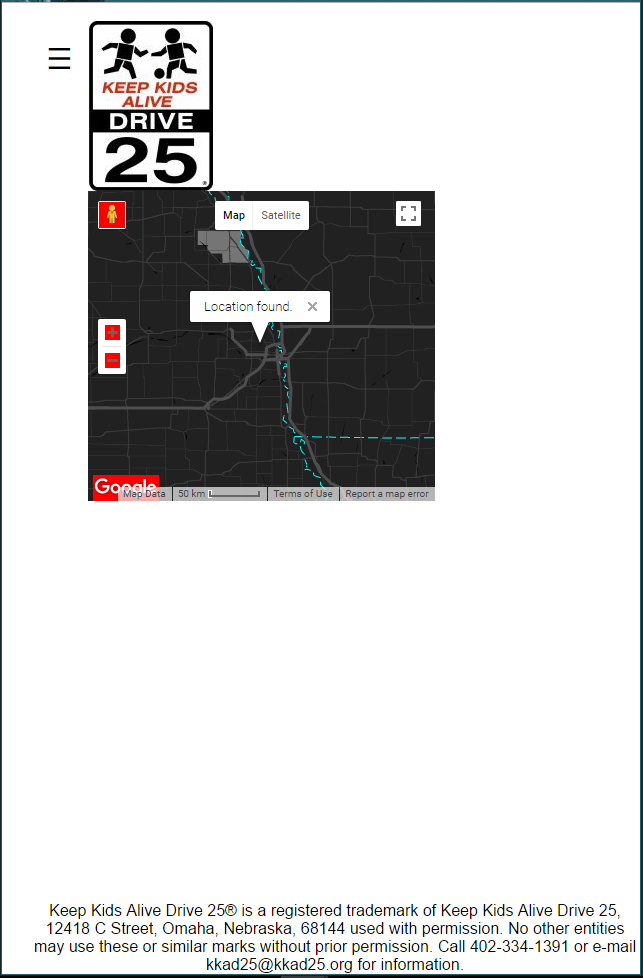
1. The Map Page

The Map Page is the main way of visualizing data. It consists of Jquery mixed with a Google Maps API. The Jquery is used for ajax requests to pull the data needed from home.php and the API creates the map that allows for visualization of the data.

* 1. The Map API

The map is an intuitive way of viewing the data. Several controls have been added, including zoom and full screen functions which are outlined in the image depicting the map. The points are added via this for loop:

This for loop allows for dynamic expansion of the number of points on the map. Thanks to this, you don’t need to create new point objects every time a new data point is submitted.



Full screen controls.

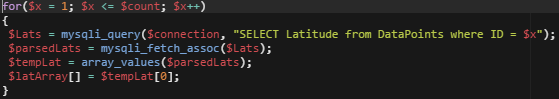
Zoom controls.

This map can be set to full screen using the full screen button, and you can zoom or un-zoom the map using the zoom controls. The three lines on the top is the main navigation menu. For further details regarding the navigation menu, please see section 5.1.

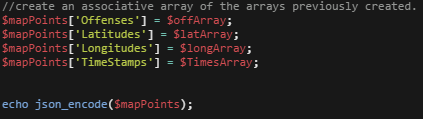
* 1. Home.php

The data is acquired from the data table in the sql server using the home.php file. This file again uses for loops to create dynamically expanding arrays of data to send to the pages that reference it:

Column detailing the date and time related to each submittal.



This is done for the latitudes, longitudes, offenses, and timestamps present in the table. It also packages all of the created arrays into a JSON object for easy transport to the pages that need the data:

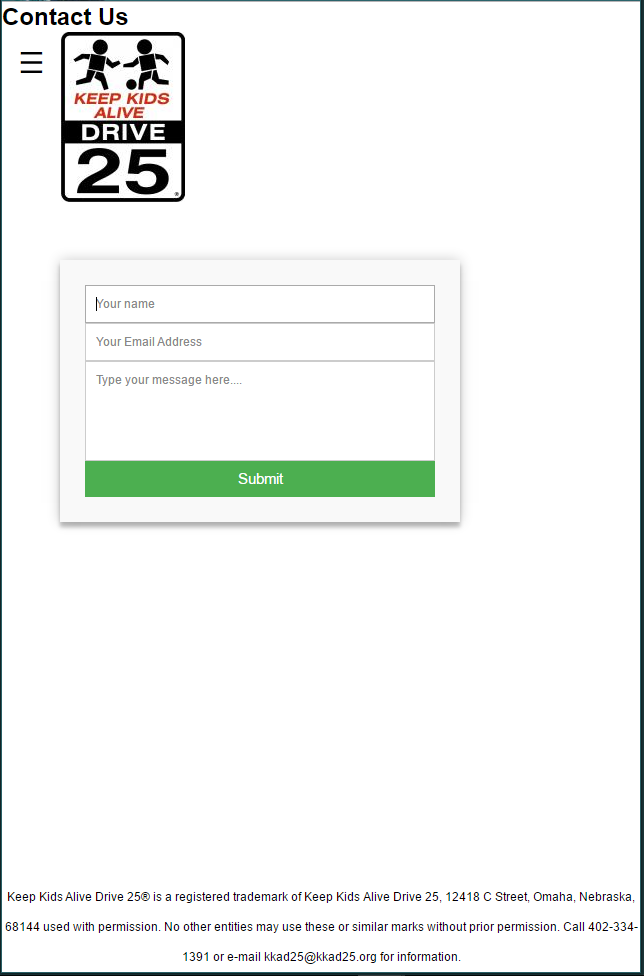


Which the pages decode and use as an associative array of arrays containing the data from the php table.

1. The Contact Page

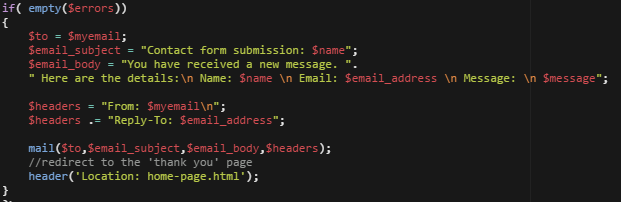
The Contact page makes use of the contactus.php file to send an email to a specified adress containing the senders email adress, their name, and the message they wish to send.

* 1. The Contact Page and the Form



Submittal form.

The page uses a simple submittal form to post the data to contactus.php. upon clicking submit, provided that all fields are filled out, the user is redirected to contactus.php, which gets and processes the data and sends it using the mail() function:



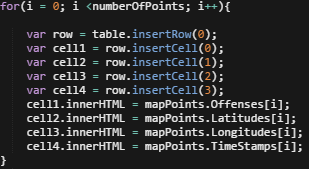
It then redirects the user back to the home page. The myemail variable is defined earlier, and contains the email that the message is sent to. Change this if to send the email to someone else. Otherwise, this process is completely automated.

1. About page

This page is relatively simple, consisting of only one paragraph of text in a <p> tag, and the styling required for the navigation menu. Change the text in the paragraph tag to change the text on the page.

1. The Table Page
   1. The Construction of the Table

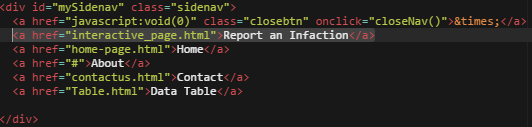
The table is constructed using the data received from the home.php file via Jquery ajax request (see the map page section for details on how that data is structured). The table is designed with a for loop that allows for dynamic expansion:



This means that this page requires no matenance, save adding a link in the navigation bar should another page be added to the application.

1. The Navigation Menu
   1. adding additional links to the navigation bar

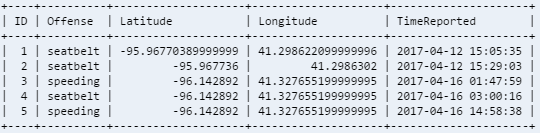
The navigation bar is a pop in and out menu that can be activated by clicking the icon on any page of the application. The process for adding a link to the list is very simple:



In order to add another page to the list, find this div tag on each of the pages that have the bar (every page should have an identical div tag). Then ad another a href tag pointing towards your new page, as well as some text inside of the tag describing what the page is. After that, it should show up in the bar wherever you put it in sequence, so if you put it between “home” and “about” in the code, it should show up between those two links in the menu.

1. The SQL Table and How it is Structured
   1. Basic Structure

When creating this table on your hosting provider, there are a few things to keep in mind. The table name is DataPoints, with exactly that casing. Should you screw up that casing, it could create connection errors. There are five columns in the table, and ID column which consists of integer values. This column is the primary key of the table, meaning that it auto increments. This is important, as several scripts used in the application rely on this column to tell how many value are in the table. The next columns are a text column called Offenses that holds the Infractions reported, two float columns called Latitude and Longitude that hold the lat and lng values for each point, and the TimeReported column which holds timestamp values for each of the points. Make sure to specify that those have to be new timestamps, so that the value can automatically be input when a new row is created.



* 1. Removing Points From the Table

Because the for loops in home.php pull values into the arrays according to the ID of the value in the table, removing values becomes a problem. The ID of the next value isn’t dictated by the ID of the previous value in the table, but rather by an internal value within the table. So if value 5 was removed another was added, that table would have values of IDs 1, 2, 3, 4, and 6. This would break the for loops creating the arrays of values. Truncating the table is a good way of reseting the ID, but it also removes all of the values in the table. If there is a way to remove single variables and setup the ID properly, use it, but this guide can offer no solution as of this revision.