Traffic Monitoring Project

Group #13
Xuan Li
Chih-Ting Cho
Ting-Chieh Huang
Jonathan Hong
Jan Racoma
Kevin Cundey

Software Engineering 14:332:452
Individual Contributions

https://sites.google.com/site/452trafficmonitoring/home

http://www.jracoma.com/trafficmonitor/

Website Application Descriptions

File Name	Short Description
index.php	Main webpage UI
functions.php	PHP functions
db_connect.php	DB Connection Configuration
TrafficParser.php	Parses traffic information
TrafficDataTest.php	Test output of traffic data
WeatherParser.php	Parses weather information
theme.css	Built from bootstrap files
css/	
bootstrap.min.css	
bootstrap.css.map	
bootstrap.css	
bootstrap-theme.min.css	
bootstrap-theme.css.map	
bootstrap-theme.css	
fonts/	
glyphicons-halflings-regular.woff	
glyphicons-halflings-regular.ttf	
glyphicons-halflings-regular.svg	
glyphicons-halflings-regular.eot	
js/	
bootstrap.min.js	
bootstrap.js	

Full Descriptions

index.php

Main website homepage.
The top area will contain links to access alerts, gas, and register/login links to their appropriate areas.
The main area is split into two sections, the graphical map and a form.
The form allows the user to specific starting location, destination location, departure time, departure date, and the option to set alerts. Fault checks have been implemented on certain fields. Both locations are restricted using jQuery to be inside the coverage area. Checks are done when the user moves out of the input field and when the "Submit" button is pressed. The departure time is set to be in 10 minute increments throughout the day. The departure date is limited to ten days in the future. When the "Set Alert" button is checked, the "Alert Method" field is enabled allowing the user to select whether they would like an alert by "Email Address" or "Phone Number." Checks are then used on each box to verify a correct email address or phone number is used. Again, all these checks are done using jQuery.
Once the user inputs all correct information, the page will interface with the Google Maps API using JavaScript and request directions from starting location to destination location. Inside the request, an option to retrieve alternate routes is enabled. Google Maps API then returns a JavaScript object containing the legs of the route, duration, anticipated arrival time, and alternate routes. Once the results are received, the form verifies that directions were returned otherwise an error is displayed. The results are then again sent back to Google Maps API to update the map and display the route on the map. The form panel will then switch to step-by-step directions and also present the user with an option to select an alternate route, if it exists. A "Reset" button is placed if the user wishes to generate a new route.

functions.php

PHP file containing all functions used by the application.

db_connect.php

PHP file that holds MySQL database connection information and establishes connection with the database.

TrafficParser.php

Accesses:
http://www.mapquestapi.com/traffic/v2/incidents?key=Fmjtd%7Cluur216t2u%2Ca5%3Do5-

90tnlz&callback=handleIncidentsResponse&boundingBox=40.9948872,-74.50699,40.461404,-73.6813134&filters=construction,incidents&inFormat=kvp&outFormat=xml
The URL is a MapQuest API query that includes a bounding box that restricts our coverage area, types of incidents to be report, and the output format desired.

TrafficDataTest.php

Simple webpage displaying data contained in the trafficData table.

WeatherParser.php

Accesses:
http://api.wunderground.com/api/9dfec0046b8e4547/conditions/q/NY/New_York.xml
URL was generated with aid from Wunderground API for weather conditions in the New York, NY area. The weather complexity was simplified to one zip code since the coverage area will generally experience the same weather conditions.

theme.css, css/, fonts/, js/

Built with bootstrap files. To simplify design, we used a prebuilt web design and built our website on top of it. These are the files and folders that are used to generate the web page.

Database Descriptions

trafficData Database Schema:

Field Name	Type	NULL	Default
id	int(10)	NO	0
lat	decimal(10,6)	YES	NULL
lng	decimal(10,6)	YES	NULL
type	int(1)	YES	NULL
severity	int(1)	YES	NULL
startDate	date	YES	NULL
day	varchar(9)	YES	NULL
startTime	time	YES	NULL
endDate	date	YES	NULL
endTime	time	YES	NULL
shortDesc	varchar(255)	YES	NULL
fullDesc	varchar(255)	YES	NULL
distance	decimal(4,4)	YES	NULL
delayFromTypical	decimal(4,4)	YES	NULL
delayFromFreeFlow	decimal(4,4)	YES	NULL

weatherData Database Schema:

Field Name	Type	NULL	Default
recordDate	date	NO	0000-00-00
recordTime	time	NO	00:00:00
weather	varchar(255)	YES	NULL
temp	decimal(3,1)	YES	NULL

userDatabase Database Schema:

Field Name	Type	NULL	Default
username	varchar(255)	NO	None
password	varchar(255)	NO	None
emailAddress	varchar(255)	NO	None
phoneNumber	varchar(255)	YES	NULL

alertsDatabase Database Schema:

Field Name	Type	NULL	Default
id	int(11)	NO	None
username	varchar(255)	NO	None
alertDate	date	NO	None
alertTime	time	NO	None
sLocation	varchar(255)	NO	None
dLocation	varchar(255)	NO	None
alertMethod	varchar(255)	NO	None