DRIVER User Manual

Web Application

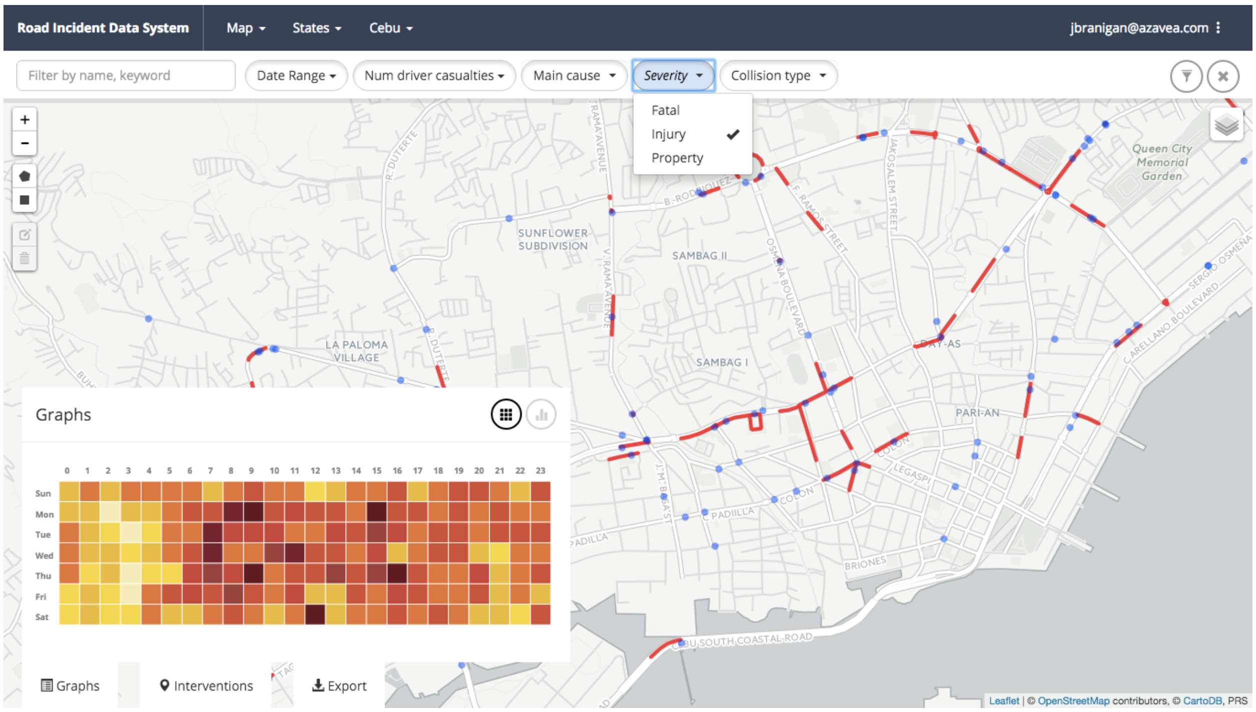


Table of Contents

Introduction

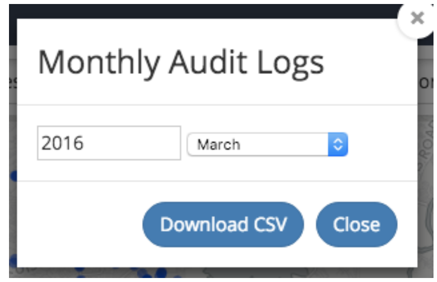
This document provides usage instructions for the DRIVER application. DRIVER is designed to collect and analyze data about incidents that occur at a particular place, such as traffic accidents or crimes. The application provides data entry tools, graph and map interfaces, custom report and filter tools, multiple concurrent user editing, and data exports.

This is a draft document and is subject to revisions.

Notable items pending completion in this document:

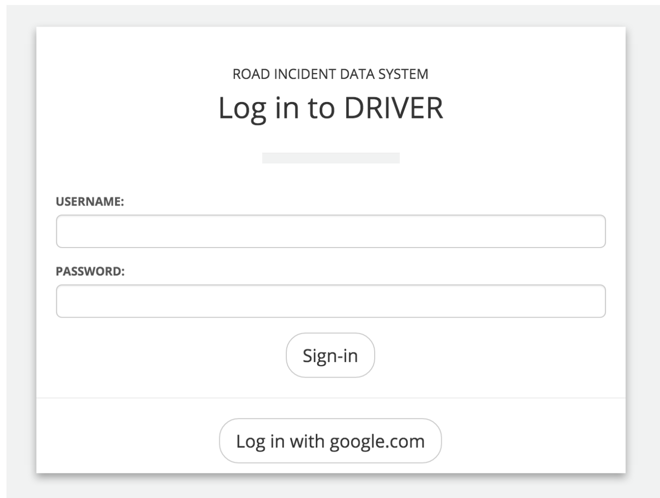
- Inclusion of acknowledgements section
- Completion of statistical model documentation
- Completion of custom reporting interface documentation

Authentication and User Management



DRIVER provides two options for logging in to the application: a username and password pair, or single sign-on (SSO). Multiple SSO providers may be configured in the software, and it comes by default with Google account integration.

Single Sign-On



If a user has an account with a supported provider, the can click on the “Log in with ‘provider’” button. This process redirects to the provider’s authentication page, and after a successful login, redirects back to DRIVER with the user’s credentials.

Username and Password

Administrators may create username and password pairs for users to log in. This approach requires an administrator to be available for user registration support.

User Roles and Permissions

There are three roles with differing permissions in DRIVER. Roles and permissions were defined in order to provide different levels of functionality to different types of users.

Admin

The admin role has access to the most functionality in the application, including login capability to the database design editor software. Admins can modify the structure of the database, add new fields, make fields required, upload geographic boundaries, and manage users. The database design editor software will be covered in a separate user manual. In DRIVER, admins may export user access logs for analysis.

Analyst

The analyst role includes permissions to view and edit all data in DRIVER. Analysts may add, edit, and delete incidents and interventions, and have access to all event information, including details associated with an incident such as people and vehicle information.

Public

Public users may register to view basic incident data. They may not edit any data, and can not view person or vehicle information.

Viewing Access Logs

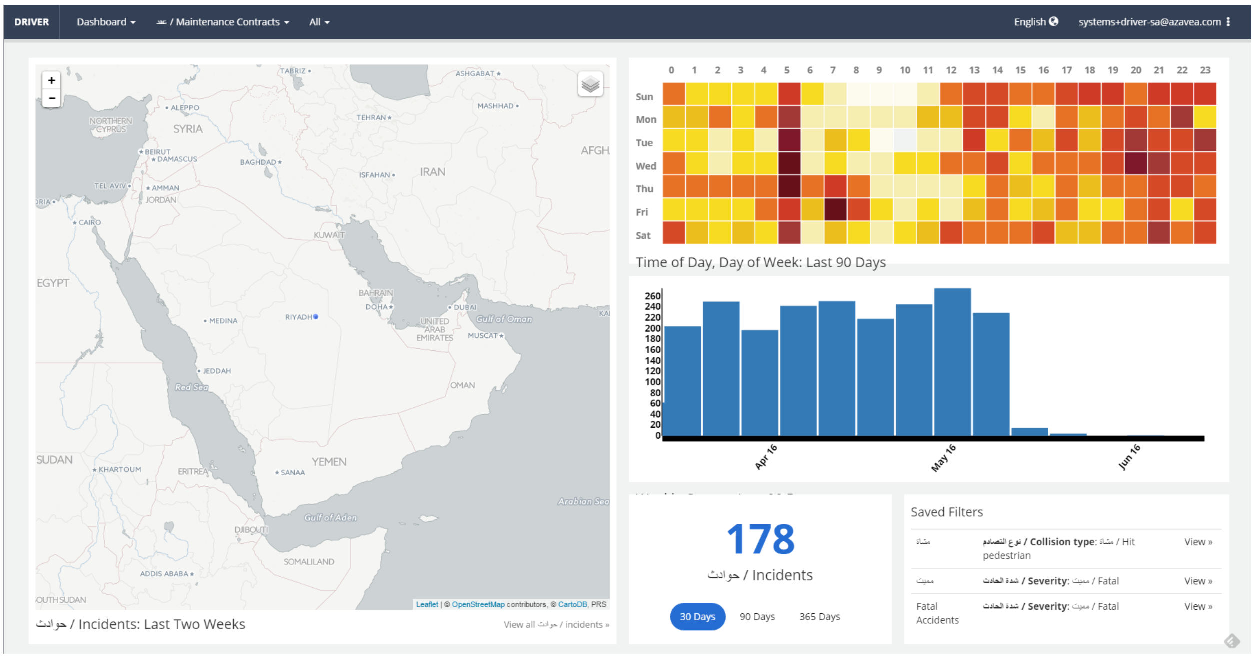
Admin users can click on their username in the top right of the screen to view access logs. A modal window will appear that allows the user to choose the month and year of access logs to download. Logs include the date and time an incident was added or modified, the username, and a link to the incident view page.

View Interfaces

There are three primary interfaces in DRIVER, each of which provide different ways to analyze and view incident data. The three interfaces are linked in different ways and rely on options chosen in the header that is visible on every interface. The map and record view interfaces rely on options shown in the filter bar, which is immediately beneath the header.

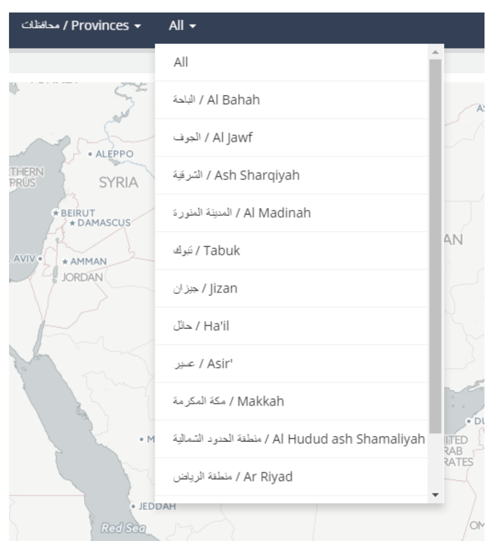
Dashboard

The dashboard interface is the default view after logging in. This interface includes summary and overview information about the data in DRIVER.



Application Defaults

The dashboard has several default settings to provide a quick analysis of recent events. The Time of Day, Day of Week (ToDDoW) chart shows the concentration of incidents over the last 90 days. There is also a map of accidents that occurred during the past two weeks, and a map of current black spots, color coded by severity. Two widgets in the top right corner display statistics and totals for sets of time periods.

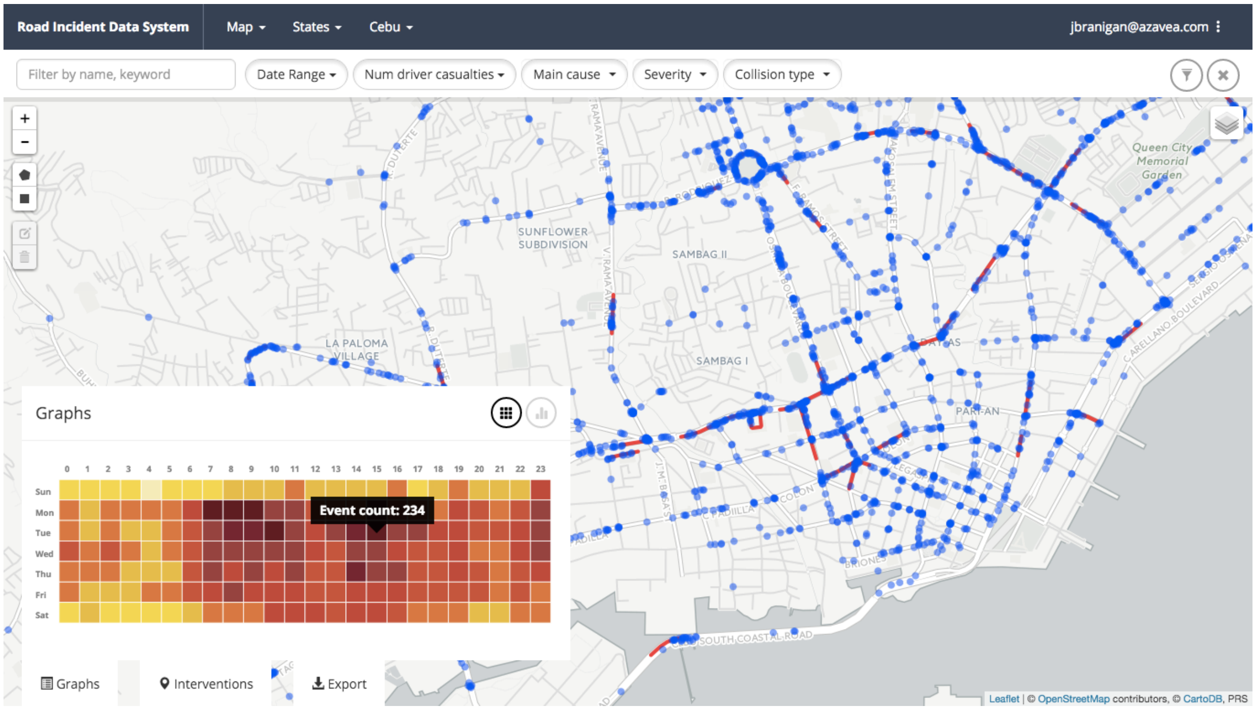


Region Default

A user may set their region default in the header bar. This could be a region in the country, or any smaller subset according to geographic boundary data loaded into the application. The region chosen will filter the maps, ToDDoW chart, and widgets to provide data for incidents that occurred only within that geographic boundary. This default setting is preserved between user sessions on the same computer.

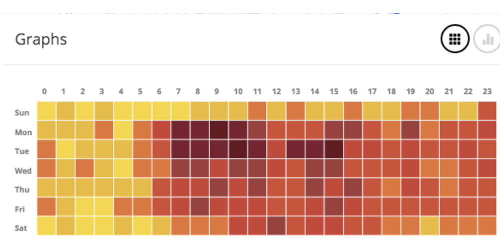
Map

The map interface is one of the most data rich views in the application. It contains a variety of layers, a full set of available filter tools, and access to interventions, additional graphs, custom report builder tools, and data exports.



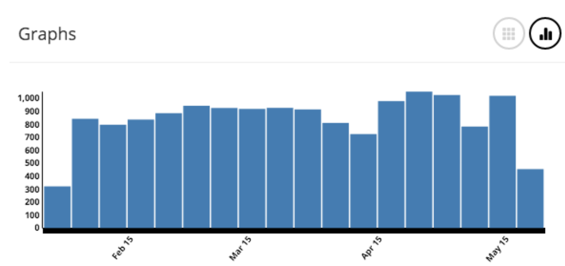
Layers

Toward the top right of the map pane, there is a layer switcher tool. DRIVER shows a default list of native layers, and includes additional geographic boundary layers as they are loaded into the application. The default base map is drawn from OpenStreetMap data. There are both point and heat map style layers to represent incidents.



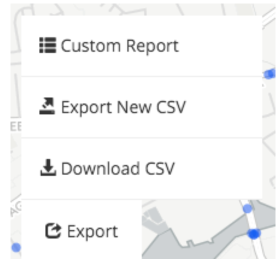
Graphs

At the bottom left corner of the page, there is a tab that expands to display graphs. Users may toggle between graph styles using the button at the top right of the chart. The data loaded into the graphs is controlled by the currently applied date range and filter set, including geographic filters.



There are two types of graphs. One is a ToDDoW graph similar to the one on the dashboard. The only difference is that the data is filtered according to what the user sets. The second graph is the total number of incidents over time, aggregated by week. Both graphs provide interactivity on mouse hover.

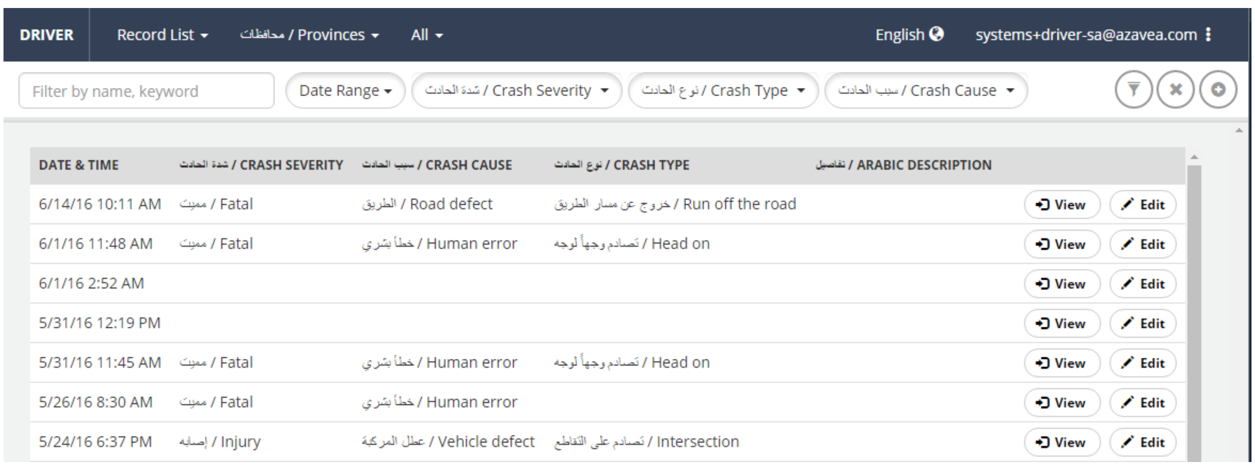
Exports



Data in the map may be exported as a set of CSV files. DRIVER data is stored in a relational database that supports many-to-one relationships. This means that for a given incident, there may be many people and vehicles involved. The export CSV zip file contains a CSV for each type of related object, linked to the primary accident table via a unique ID. The CSV files also contain the latitude and longitude for the incident location. These files may be further analyzed using spreadsheet or database software.

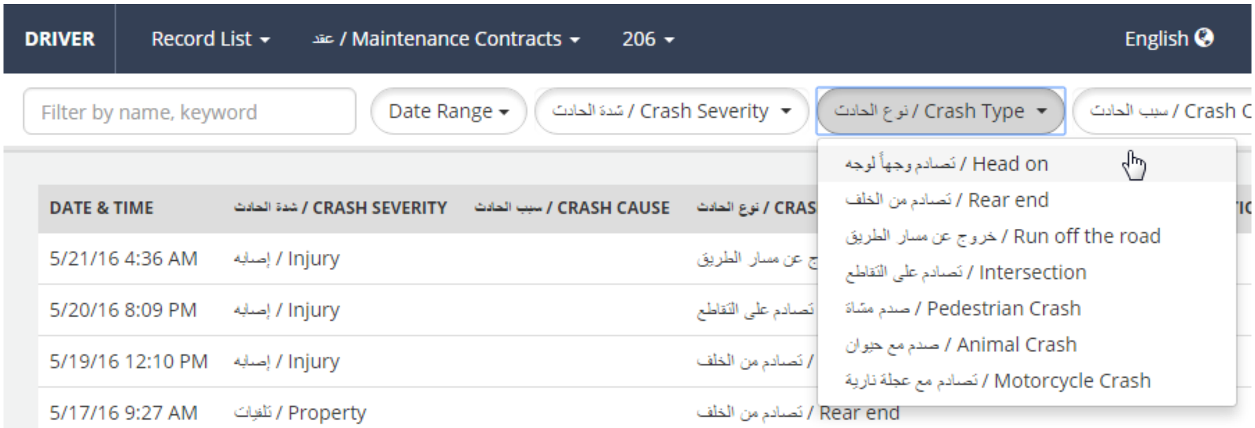
Record List

The record list interface is simply a tabular view of what data is shown in the map interface. The filter bar persists across both views, and its state is maintained when switching between the map and record views. As users update filters, the records will update accordingly. From the record view, users can view more details about an incident, or edit the incident data.



Filtering and Searching

DRIVER provides a variety of ways to interact with the map and list view by using filter and search tools to analyze subsets of the incident data.

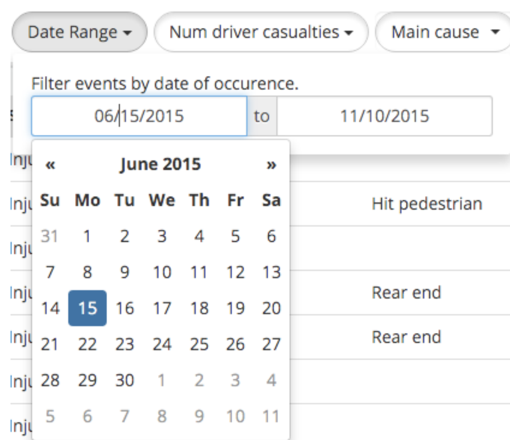


Attribute Filters

DRIVER syncs with the database designer schema to make filterable fields available to users. Fields marked as filterable/searchable are included in the filter bar.

Text Search

Searching in the text box will narrow down the incidents to ones that include the search term in a field marked as filterable/searchable in the database designer. Typing multiple words will perform a search for incidents that have both of the words in one or more fields.



Date Filter

The date filter has a clickable interface to choose minimum and maximum dates for display and analysis. The default range is the previous ninety days.

Option Filter

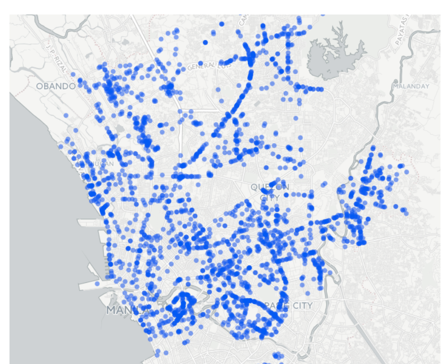
Enumeration fields (select lists, checkboxes) are included as attribute filter dropdown buttons. Selecting an option applies that filter to the data. Multiple options may be selected for any filter. Active filters are highlighted in light blue in the interface.

Numeric Filter

Fields containing numbers may be filtered using a minimum/maximum range in the numeric filter.

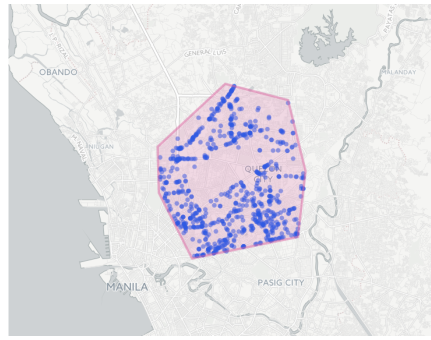
Spatial Filters

DRIVER employs two levels of spatial filtering. The first is a geographic boundary filter, which is chosen in the page header, and the second is a custom boundary drawn by the user.



Region Filter

When a geographic boundary is selected in the header, the map zooms to the extent of that boundary. Additionally, data is filtered to what is contained in that boundary. For example, incidents from Cebu would be filtered out if the user selects the National Capital Region. This filter is applied on the map, record, and dashboard pages.

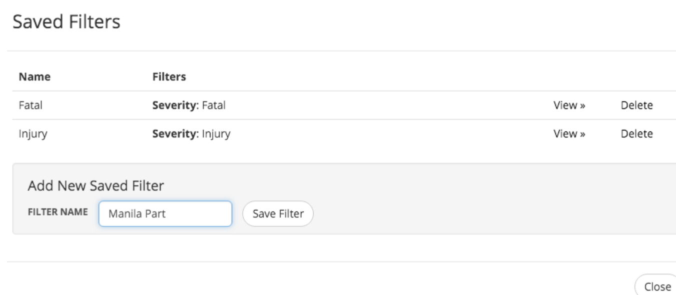


Custom Area Filter

Users may draw a custom boundary to filter down to incidents that occurred in that area. The boundary drawing tools are on the left side of the map, and include draw (rectangle or polygon), edit, and delete buttons. A tooltip provides instructions while interacting with the tools.

Saving Filters

Applied filter sets can be saved for later use. The saved filters interface allows users to save and name the currently applied filters, and lists previously saved filters. This can be useful for defining a combination of filters used to create reports, exports, or other analysis tasks. Saved filters do not apply to geographic boundary selections or date ranges, but act on search, option, and spatial filters.



Clearing Filters

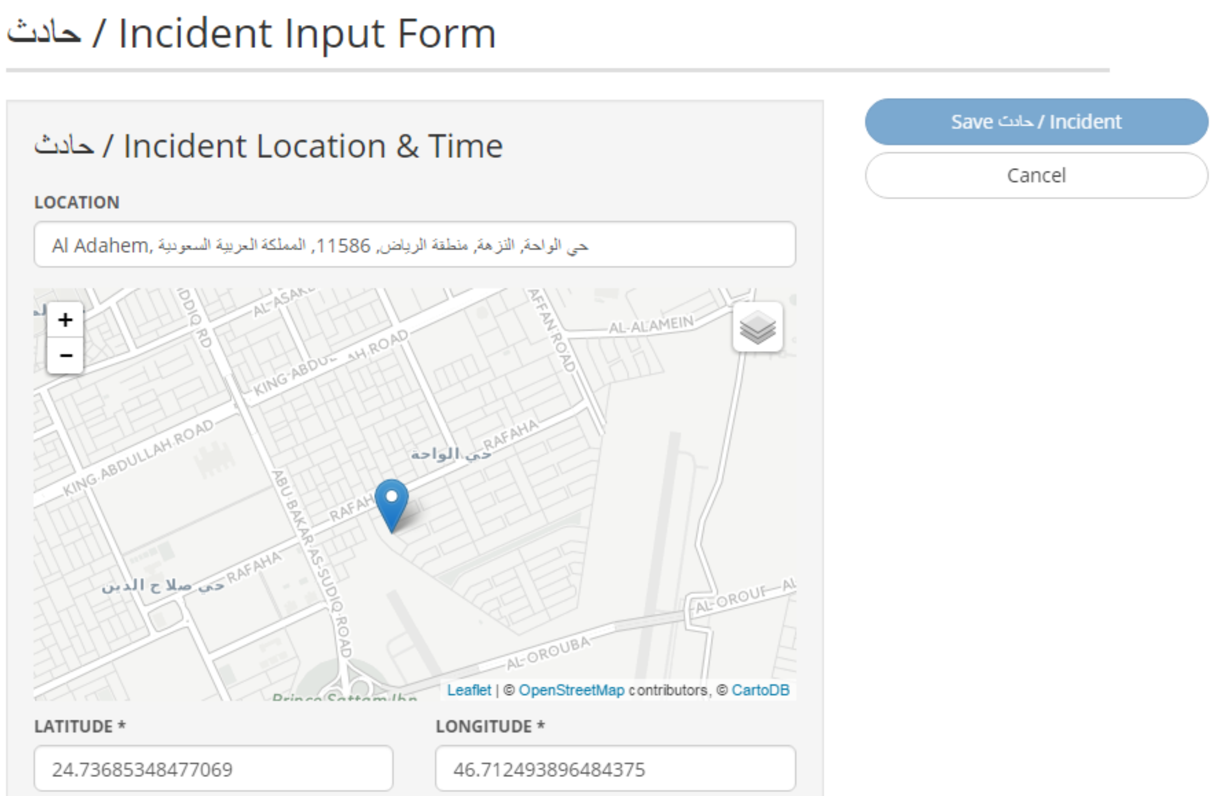
Filters may be reset to default state by clicking the clear filter button.

Data Entry

A fundamental aspect of DRIVER is to collect and store incident data, and the application provides a simple data entry interface, both on desktop and Android devices.

Web Interface

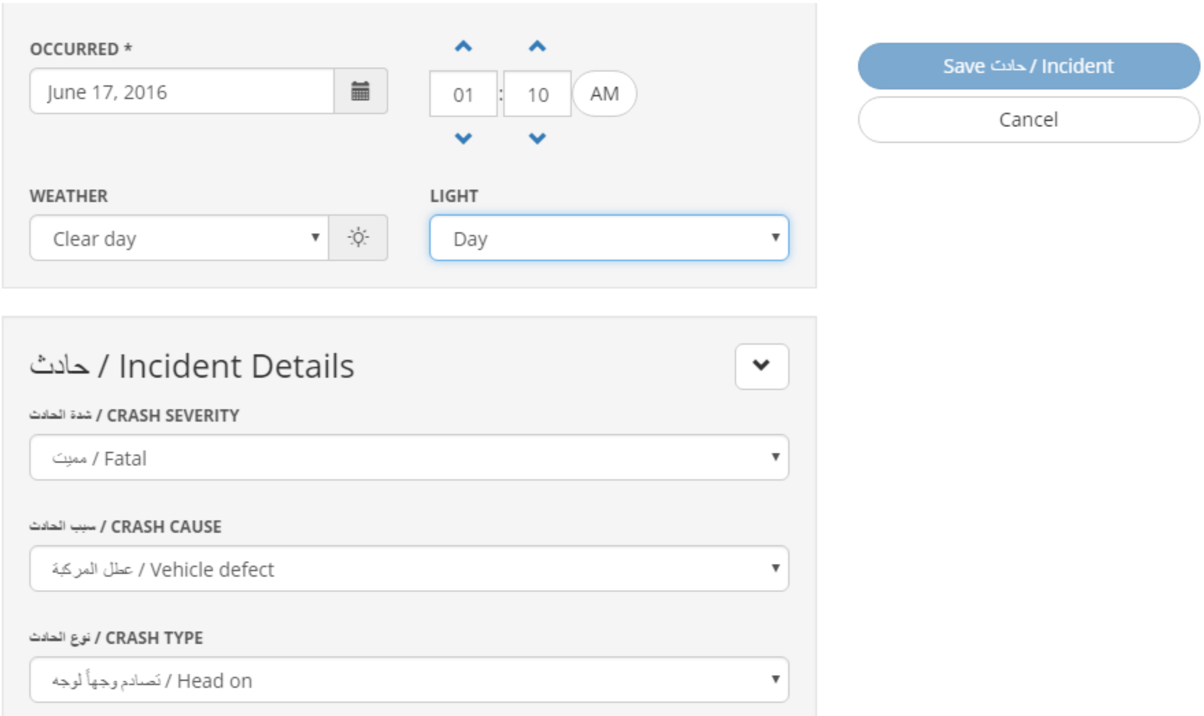
To add or edit incident data, a user must have analyst or admin privileges. Begin by clicking the 'Add New Record' button on the right side of the filter bar. A save and cancel button are fixed on the right side as the user scrolls down to complete the form.



The location search at the top of the form autocompletes street addresses and place names from OpenStreetMap. Users may also select a location by clicking on the map.

Weather and light fields in the form will use the forecast.io API when the data is saved, if the fields are left blank. Users may choose to fill in the fields and not use the API.

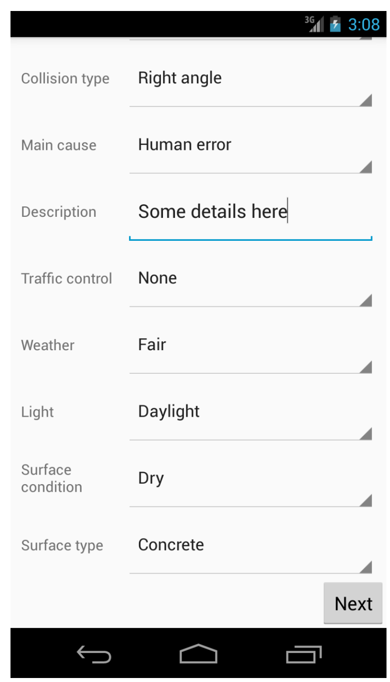
Required fields will be highlighted in red if they are not completed before saving the data.



Android Application

An Android application was built for data collection in the field. This application does not require a constant data connection in order to work, and will sync to the central application database when a connection is acquired either on a cellular network or on wifi.

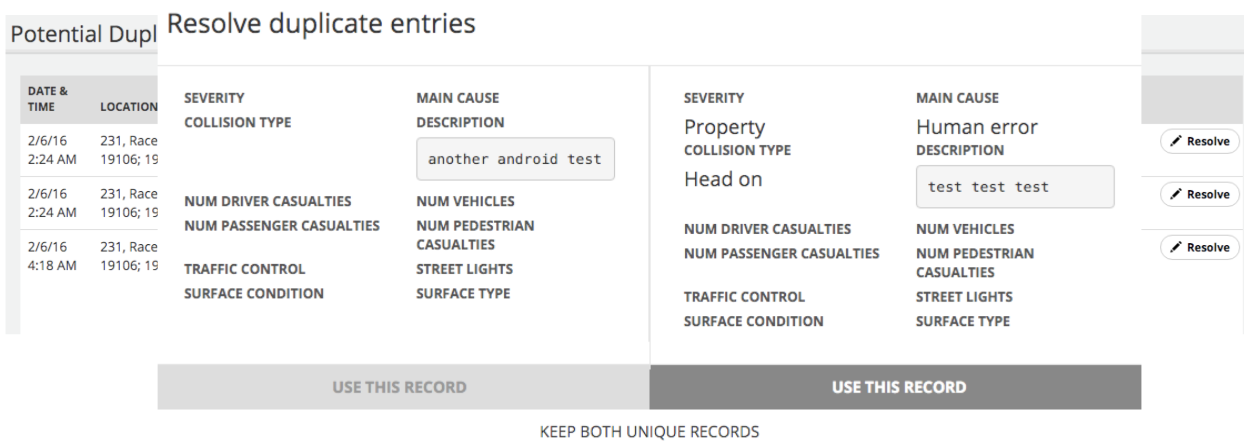
The application relies on the GPS chip in the device. Upon opening the app, it will begin to connect to GPS to get the current location. Data entry for incidents should happen at the location of the incident for most accurate results. Locations may be edited in the full web application after incident data is synced.



Duplicate Record Management

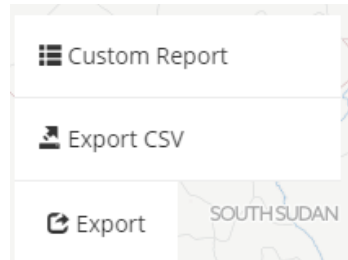
Occasionally, two users may input the same incident. DRIVER has a process that runs on record creation and nightly to identify potential duplicate incidents. Admin users may access an interface to identify and resolve the duplicates.

After choosing to resolve a potential duplicate, the users are presented with data from each record, and may choose to use one or the other, or keep both records.



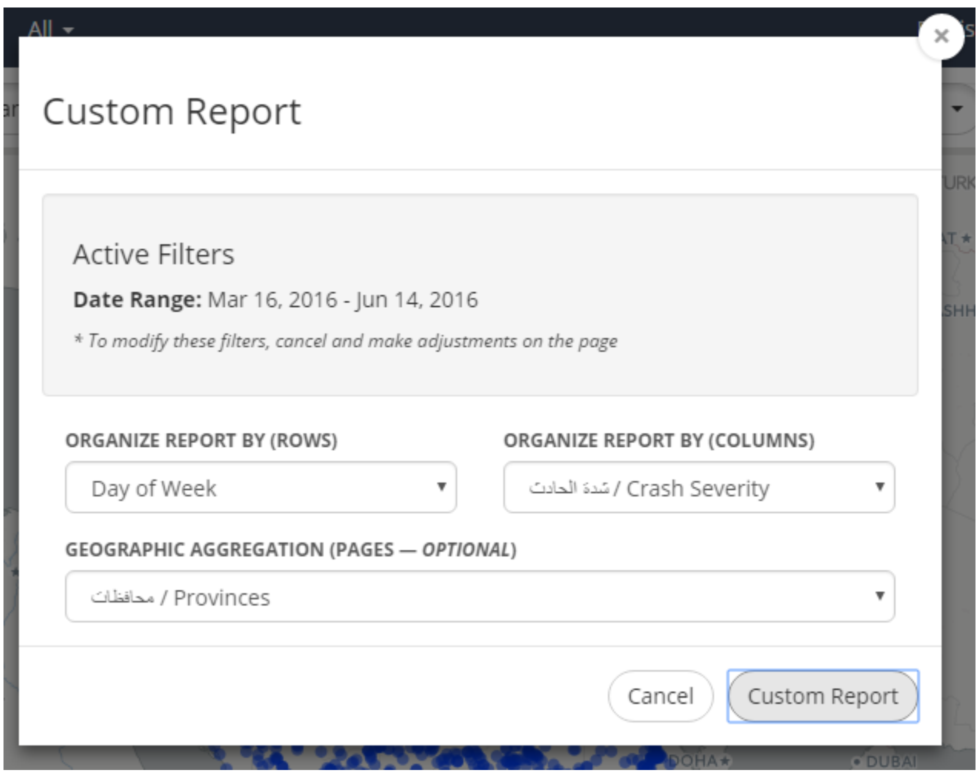
Reporting

The DRIVER application includes functionality to generate custom reports from the database. These reports aggregate counts of incidents according to the rows and columns defined by users generating the reports. The reports also allow for geographic aggregation based on any uploaded boundary layers.



Reports are generated from the map page by clicking Export > Custom Report.

The user is presented with a pop-up window that provides the option to select rows and columns from the data fields defined in the schema. The user is also able to specify a geographic aggregation to break the counts of incidents out by the regions of a boundary layer. Note that any filters that are active for the application will apply to the results of the report.



Upon clicking 'Custom Report' the user the report is generated in a separate window for the user to view. The results of the above selections can be seen in the report below.

DRIVER | Custom Report ▾ | محافظات / Provinces ▾ | All ▾ | English | systems+driver-sa@azavea.com

Mar 16, 2016 - Jun 14, 2016

حوادث / Incidents for مكة المكرمة / Makkah by Day of Week and شدة الحادث / Crash Severity

DAY OF WEEK	ميت / FATAL	إصابة / INJURY	تلفيات / PROPERTY	TOTAL
Monday	0	51	14	66
Tueday	0	40	10	50
Wednesday	1	45	8	54
Thursday	0	56	15	71
Friday	1	58	19	78
Saturday	1	46	18	65
Sunday	2	50	18	70

Mar 16, 2016 - Jun 14, 2016

حوادث / Incidents for جيزان / Jizan by Day of Week and شدة الحادث / Crash Severity

DAY OF WEEK	ميت / FATAL	إصابة / INJURY	تلفيات / PROPERTY	TOTAL
Monday	0	9	2	11
Tueday	0	12	1	13
Wednesday	0	13	5	18
Thursday	0	11	2	13
Friday	0	9	6	15
Saturday	0	15	5	20
Sunday	0	16	3	19

As selected, 'Day of Week' serves as the rows of the custom report and 'Crash Severity' serves as the columns. Counts of incidents can be seen in each cell and the sum in the total column to the right. Additionally, a geographic aggregation by Province was selected so individual reports have been generated for each province. The data in each report has been filtered by each of these geographic boundaries.

Black Spot Identification

Black spots are sections of roadway or intersections that experience an unusually high number of traffic crashes. Black spot locations are calculated on a daily basis with a server-side R script. The script uses a gradient boosted model (GBM) to predict severe and non-severe crashes in traffic segments based on geography and historical severe and non-severe crashes. Black spots require a minimum of one year of consistent data to produce the best results.

Modeling

This section explains inputs and outputs of the black spots model. The black spots model ingests the data below and predicts crashes in road segments. The purpose of the GBM is to learn from all of the available information. One possible alternative to this type of model is a simple historical average, for example, expecting that this year, each segment will have as many crashes as it did last year. The “counting model” doesn’t have any understanding of a “surprisingly” high count, and would be very wrong any time unusually high counts or unusually low counts in a segment didn’t carry over from one year to the next.

The GBM predicts based on more than historical counts and predicts for groups of segments that are similar to each other instead of individual segments. There are three benefits to using the GBM instead of other classes of model. First, a GBM will more effectively handle regression to the mean than counting models will. Other models, for example, linear regression models, also better handle regression to the mean than counting models, but, second, GBMs more gracefully handle non-linearity and interaction effects than linear models. Finally, because GBMs have several tunable hyperparameters, the black spots prediction script can search over the possible GBMs to pick the best one.

Input Variables

Each of the variables in the following list must be included in an input CSV for the model to be able to run. Below, a “segment” refers to a section of road or an intersection and “present” refers to the final year of training data.

- inter: (binary) whether a segment contains an intersection
- length: (float) the length of a segment in the units of the projection
- lines: (integer) the number of line segments comprising the traffic segment in which the crash occurred. All segments with more than one line contain an intersection.
- pointx: (float) the x coordinate of the centroid of the segment, in the units of the projection
- pointy: (float) the y coordinate of the centroid of the segment, in the units of the projection
- t0notsev: (integer) the number of non-severe crashes in the segment in t0, where t0 is the current period, extending from a year ago to the present
- t0sev: (integer) the number of severe crashes in the segment in t0
- tXXnotsev: (integer) the number of non-severe crashes in the segment in tXX, where tXX is the period spanning from XX years before the present to one year before that. For example, t1 includes from two years before the present to one year before the present.
- tXXsev: (integer) the number of severe crashes in the segment in tXX

For tXXnotsev and tXXsev, the script will by default look for variables matching from t3 to t1 for training and will look for a t0 or t1 for results. This behavior can be modified in the PrepData function.

A preprocessing script creates these values from OpenStreetMap data and road safety records. The preprocessing step performs three functions to transform crash records and road shapefiles into the data used above.

First, the preprocessing script identifies all intersections and assigns a buffer zone around each intersection to that intersection. In cases where buffer zones from intersections overlap, those intersections can be assigned to the same segment.

Second, it breaks all stretches of road without intersections into smaller pieces. This step is necessary because identifying, for example, a 10-mile stretch of road as a black spot doesn't provide good information about where along that stretch is dangerous or whether the stretch of road has more crashes because it's a long stretch of road.

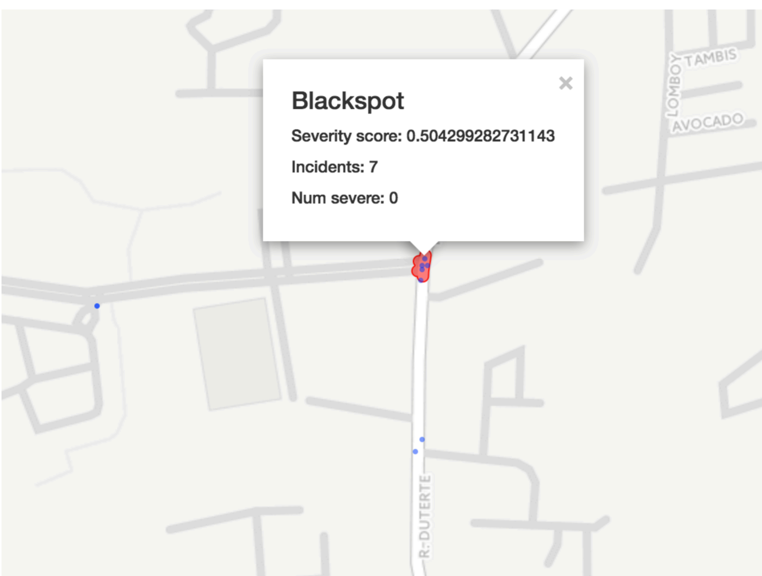
Finally, it assigns every crash record to its nearest road segment and aggregates crash counts by segment and time period.

Outcome Variable

The outcome variable can be set by the user in the PrepData function of the included script. If the user sets outcome.year to 0, the model will be trained and tested on the same year's data. If the user sets outcome.year to -1, the model will forecast future data.

While the outcomes themselves — numbers of crashes of different types — are counts, the forecasts themselves will be floating point values.

These predictions determine which black spots are available for enforcer missions. Users set a percentile threshold for which black spots to consider in the ASHLAR editor; for example, users could choose a threshold value of 0.95 to select the top five percent most severe black spots. The DRIVER application then chooses enforcer missions randomly from among the black spots that clear this threshold, giving more weight to black spots with higher severity values.



How the Model Works

This section explains the basics of decision trees and boosted decision trees.

Decision Trees

Decision trees are series of if-then statements that govern predictions about the behavior of a given system. For example, in predicting whether a home is in San Francisco or New York based on its price per square foot, its altitude, its number of bedrooms, etc., a decision tree approach may be used. Decision trees accomplish this by finding optimal rules based on the available data.

For example, consider the following data in the context of minimizing the total variance across groups in y after each split in x (we're only allowed to make one split at a time):

y	x
1	0
2	1
3	3
1	1
2	1
7	4.1
8	12
9	19
7	8
7	1

Splitting the above table at x > 3 results in the lowest total variance in the two groups; the group with x values greater than 3 includes 7, 8, 9, and 7, while the group with x values less than or equal to three includes 1, 2, 3, 1, 2, and 7. The decision tree at this point would guess the mean for each of these groups – 7.75 for the x > 3 group and 2.67 for the x <= 3 group. After this split, the tree has a depth of 1.

With the toy data above, total variance is already pretty low, but it is possible to find another split within each of the groups to decrease total variance further. With an additional split, the tree depth would be 2. As depth increases by a level at a time, both predictive accuracy increases and the predictions more closely link to the particular sample of data that we've drawn.

Boosting

Boosting a tree is the process of successively fitting many trees on the first tree's errors.

With the predictions from the simple tree above, it's easy to calculate the error, shown in the table below:

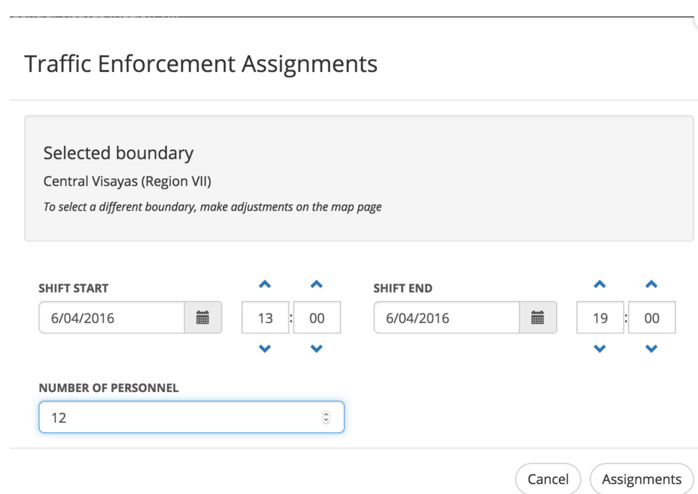
y	x
-1.67	0
-0.67	1
0.33	3
-1.67	1
-0.67	1
-0.75	4.1
0.25	12
1.25	19
-0.75	8
4.33	1

Without needing to know where those errors came from, boosting fits *another* tree to explain those errors — the errors from one tree become the target of the next tree. Each tree improves slightly on the previous tree. In sequence, what each tree provides is a better prediction of the errors from the previous tree. For additional information on boosted trees:

- This blog post includes helpful animations of improving trees in successive rounds: http://freakonometrics.hypotheses.org/19874
- These slides with a mathematical treatment include a helpful "model improvement game" starting on slide 11: http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

Enforcer Reports

Enforcer reports are designed for optimizing the deployment of traffic enforcers by focusing on the roads predicted to have the highest likelihood of traffic incidents. The reports take statistical predictions using location, time of day, day of week, and day of year, and then output a set number of locations.

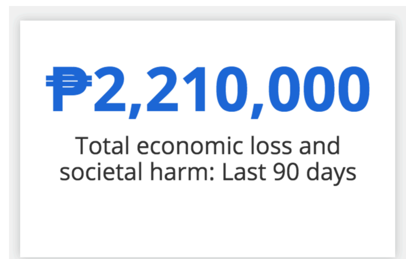


The reports are designed for printing and distribution at the beginning of a shift. An area for notes allows enforcer captains to include additional details for the shift and location.

Assignment:	**Assignment 3**
Location:	**10.310, 123.893**
Notes:	

Cost to Society

Economic loss and societal harm costs are calculated based on an Ashlar configuration. Administrators choose an option-type field associated with the primary record type, such as “Severity”, and associate a cost with each of its values.



In the DRIVER interface, this cost is displayed both in the dashboard and map views. The cost totals respect the geographic, temporal, and attribute filters applied in the map. This makes it possible for a user to associate an estimated cost for a search like *crashes involving pedestrians in a certain neighborhood in 2016*.