Emergency Resource Management System

The Emergency Resource Management System (ERMS) is an information management tool that supports government agencies and municipalities in locating and activating resources after an emergency such as a natural disaster, hazardous material spill, act of terrorism, nuclear incident, or another catastrophic event. The system is used because such events require resources above and beyond the set of resources that would normally meet a municipality's typical operational needs. Users of the system can add their own resources to the system, search for resources based on keywords or location, request available resources, track the status of resources, and record information about emergency incidences.

Users of the system may be individuals, municipalities (e.g., City of Atlanta), government agencies (e.g., FEMA, FBI, Georgia State Patrol), or companies. The following information must be managed for all users:

- **name**
- **username**
- **password**

Additionally, the system must record the **population size** of municipalities, the location of a company's **headquarters** (a single string, e.g., "Downtown Atlanta"), and the **jurisdiction** of government agencies (a single string, e.g., "Federal", "State", or "Local"). Users of the system who are individuals must have a job title and a date that they were "hired" (aka added to the system).

Because this is a prototype system, there is no interface for registering new users. Instead, user information will be loaded by the database administrator behind the scenes. During phases I and II of the project, you should define the appropriate database structures to manage the user information mentioned above. Before the phase III demo, you should populate your system with a few users of each type.

The following sections contain a functional description of the ERMS application along with some screen mockups. *The user interfaces depicted in this project description merely serve as examples to guide your thinking.* Your project's interface may look completely different and that is fine—even encouraged! For example, you might choose to split up some interfaces we have shown on a single screen into multiple screens. You might choose to use popup windows instead of refreshing the page. A complete reorganization of the user interface is acceptable as long as your application supports the same functionality as described below.

Logging In

Figure 1 shows the ERMS login screen. All users are identified by a unique username. Providing a valid username and password combination logs the user into the system. If the user provides invalid login credentials, an error message should be displayed and the user should be returned to the login screen.

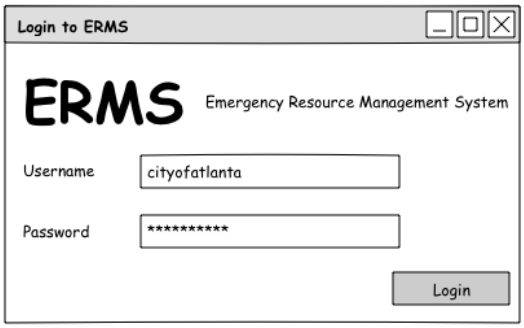


Figure 1 ERMS Login Screen

After successfully logging in, the user is presented with a menu like the one shown in Figure 2. The menu options are:

- *Add a Resource*. This allows the user to add resources that will be available for use in case of a nearby emergency incident. Other users in the system will be able to search for and request these resources.
- *Add an Incident*. The user selects this option in order to add some basic information about an emergency incident that has just occurred.
- *Search for Resources.* This option allows the user to search for and request available resources in the case of an emergency.
- *Resource Status*. This option allows the user to view currently deployed resources and manage resource requests that she has sent or received.
- *Resource Report*. This option shows a summary report of all the user's resources grouped by their primary Emergency Support Function.
- *Exit*. Logs the user out of the system and shows the login screen again.

The main menu screen should also show the user's name plus

- the **population size**, if the user is a municipality,
- the **jurisdiction**, if the user is a government agency, or
- the location of the **headquarters**, if the user is a company.

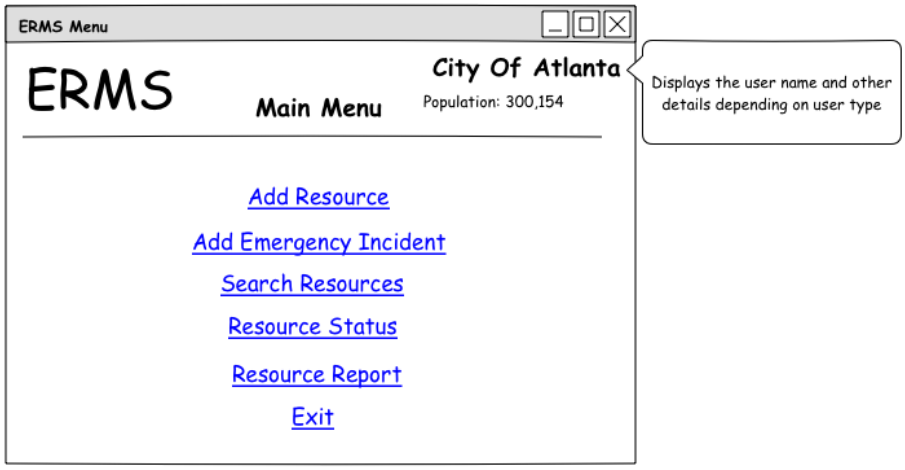


Figure 2 Main Menu

Adding Resources to ERMS

One of the primary functions of ERMS is to keep track of all kinds of resources that will be helpful when an emergency situation arises. The system encourages local municipalities, government agencies, companies, and individuals to list their resources in the system so that others can search for them based on criteria such as the resource name, function, capabilities, and home location.

When the user clicks *Add a Resource* from the main menu, a form appears like the one shown in Figure 3. Each resource is assigned a unique numerical **ID** automatically when the resource is saved. The resource's **owner** is also set automatically to the logged in user. In addition to the ID and owner, the system should maintain the following fields about resources:

- **Resource Name**. This is an arbitrary string entered by the user to describe the resource.

- **Primary ESF.** The United States federal government has defined a standard set if Emergency Support Functions (ESFs; listed below). Every resource has exactly one primary ESF. Your system should be preloaded with the following ESFs, although they should NOT be hard coded into the application so that they may be changed easily later. Each ESF has a unique number (1-15) and a brief description.

Emergency Support Functions (ESFs)

The standard list of ESFs is defined by the Federal Emergency Management Agency (FEMA). More info can be found at: http://www.fema.gov/emergency/nrf/

- #1 – Transportation
- #2 – Communications
- #3 – Public Works and Engineering
- #4 – Firefighting
- #5 – Emergency Management
- #6 – Mass Care, Emergency Assistance, Housing, and Human Services
- #7 – Logistics Management and Resource Support
- #8 – Public Health and Medical Services
- #9 – Search and Rescue
- #10 – Oil and Hazardous Materials Response
- #11 – Agriculture and Natural Resources
- #12 – Energy
- #13 – Public Safety and Security
- #14 – Long-Term Community Recovery
- #15 – External Affairs

- **Additional ESFs.** It is likely that many resources will fulfill multiple Emergency Support Functions. Therefore, the user is able to optionally assign multiple other ESFs in addition to the primary ESF. The primary ESF should not also appear as an additional ESF.

- **Model.** This field represents the model name and/or number of the resource. This is helpful to precisely identify the resource. This field is optional.

- **Capabilities.** Zero or more capabilities can be added to describe what the resource can do. Due to the wide range of possible resources in the system, capabilities are not selected from a predefined list—the user can enter any set of string values to describe a resource's capabilities.

- **Home Location.** The system records the home location of the resource in terms of latitude/longitude coordinates. Coordinates should be stored in signed decimal degrees where a negative indicates west or south. This information is important for

locating resources that are near an emergency incident. Most online mapping tools provide latitude/longitude coordinates[1].

- **Cost/Cost Per.** The system also maintains information about how much the resource costs for use. The cost is stored in dollars per hour/day/week/each. This list should be extensible without requiring code changes (i.e., it should be maintained in the database).

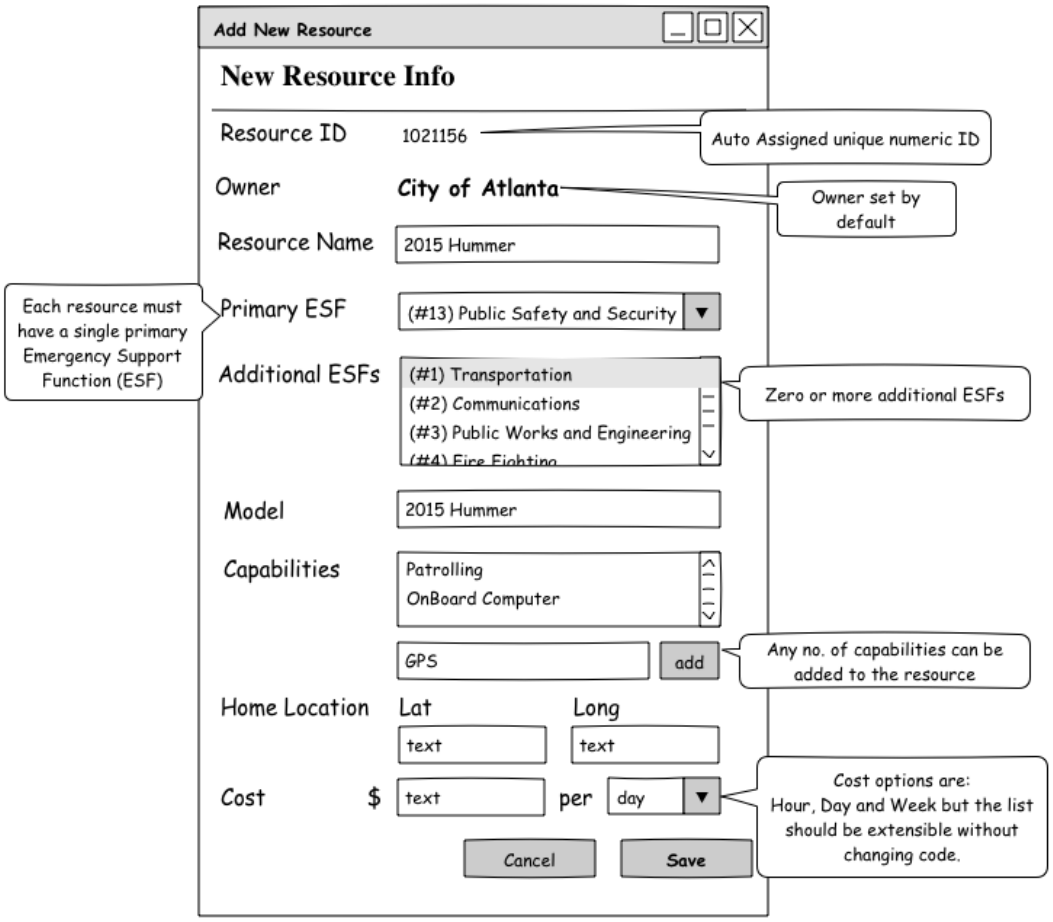


Figure 3 - New Resource form

When the user clicks save, ERMS should validate all fields before storing the resource to the database. In particular, the system should verify that all required fields are filled it, that the dollar amount is not negative, and that the latitude and longitude fields contain valid coordinates.

[1] For example, Google Maps has a Map Lab called "LatLng Tooltip" that displays coordinates of the mouse pointer when you hold down the SHIFT key.

Adding Emergency Incidents to ERMS

ERMS also allows users to track basic information about emergency incidents as soon as they occur. Figure 4 shows the *New Incident* form. Each incident is automatically assigned a unique numerical **ID** on save. Additionally, the **owner** of the incident is automatically set to the current user even though no owner field appears on the screen. The user also records the **date** of the incident, a brief **description**, and the **latitude/longitude** coordinates of the incident. If an incident is widespread, the user should choose a central location as the location of the incident (i.e., it is not possible within ERMS to define the exact boundaries of an incident).

All fields are required and should be validated before saving the incident to the database.

All incidents are private to the current user and (unlike resources) cannot be shared. Therefore it is possible that multiple users will have an incident with the same (or similar) names if the incident is widespread affecting many municipalities. For example, there may be multiple incidents named “Hurricane Lydia” in the system, but each will have a different owner.

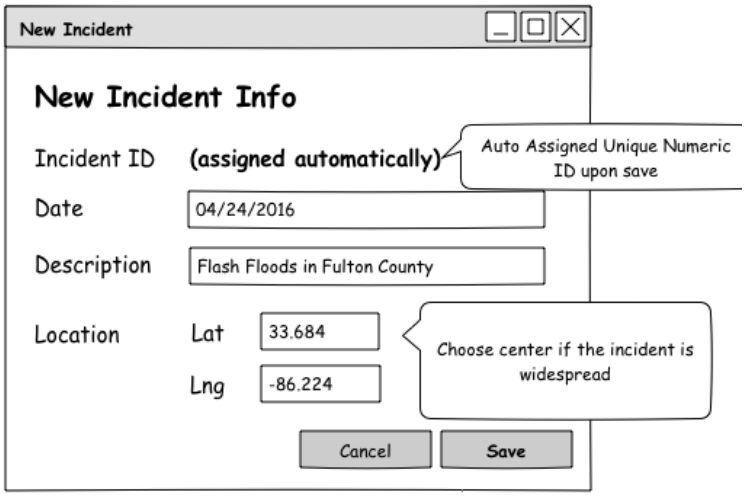


Figure 4 - New Incident

Searching for Resources

In the event of an emergency, users will search ERMS for resources that are needed to respond to the emergency. The search function returns resources that meet the user’s criteria regardless of who owns the resources (i.e., some of the search results may be owned by the user and others may not). The user may search by keyword (which includes the resource name, model, and capabilities), by Emergency Support Function (which includes both primary and additional ESFs), and by proximity to an emergency incident. The search form is shown in Figure 5.

It is possible that the user will leave all search fields blank. In this case, ERMS will return all resources currently in the system. The location-based search option allows the user to find resources within a certain radius (in kilometers) of an emergency incident that has been entered into the system. The radius of the search is a parameter provided by the user.

The *haversine* formula can be used to calculate the distance between two points defined by their latitude/longitude coordinates[2]. Given two points (lat1, lon1) and (lat2, lon2), the distance *d* between them can be calculated as follows:

$\Delta lat = lat2 - lan1$ // must be converted to radians

$\Delta lon = lon2 - lon1$

$a = \sin^2(\Delta lat/2) + \cos(lat1) * \cos(lat2) * \sin^2(\Delta lon/2)$

$c = 2 * \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$

$d = R * c$

where R is earth's radius (use 6,371 km) and all angles are in radians

Some helpful hints for calculating distances in SQL:

- MySQL provides the following mathematical functions[3] which will be useful for implementing the haversine distance formula in SQL queries: SIN(), COS(), ATAN2(), RADIANS(), and SQRT().
- MySQL allows you to create variables in a SQL SELECT clause that can be referenced in other parts of the SELECT. Doing so will simplify the SQL required to calculate distances. For example:

```
SELECT @A := COLUMN1 + COLUMN2, @B := @A * 2
FROM TEST_TABLE
```

[2] Haversine distance formula found at http://www.movable-type.co.uk/scripts/latlong.html

[3] The list of math functions supported by MySQL can be found at http://dev.mysql.com/doc/refman/5.6/en/mathematical-functions.html.

In the above SQL statement, the variable @A is calculated for every row. The variable @B refers to @A in its calculation. Therefore, the result in @B will be (COLUMN1 + COLUMN2) * 2.

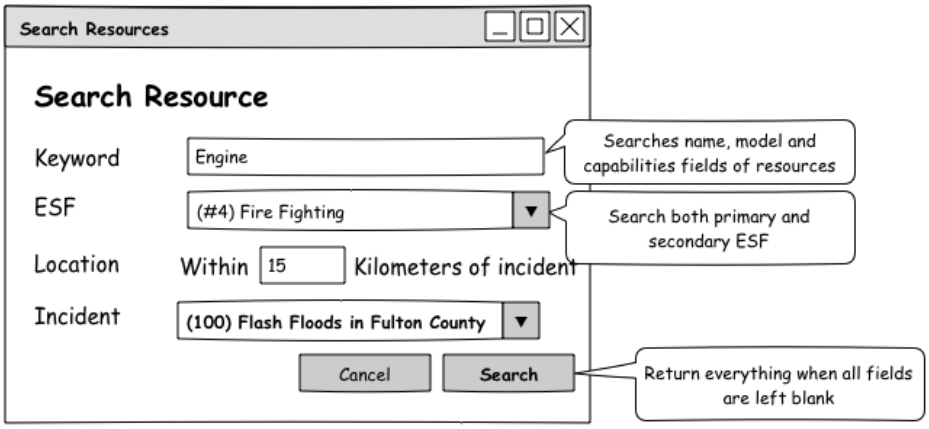


Figure 5 - Search for Resources

The search results screen is shown in Figur. In all cases, the search results screen should list for each resource the **ID**, **name**, **owner**, **cost**, and whether the resource is currently **in use**. Only those resources that match the search criteria should be listed. The keyword field searches for matching substrings in the **name**, **model**, and **capabilities** fields. If an ESF is selected, then only resources with a matching **primary** or **additional ESF** should be shown. Finally, if a location-based search is requested, then only resources with a **home location** within the given radius should be shown. Multiple search criteria should be ANDed together. For example, the search shown in Figure 5 should return resources that have the keyword "Engine", have the primary ESF of "Firefighting," and are within a 10 kilometer radius of the "Explosion at Fulton County Paper Mill" incident.

If an incident was selected in the search screen, the two additional columns appear: the **distance** of the resource (**home location**) from the incident and a column of action buttons. The search results should be sorted first by the distance with the shortest distance appearing first (if an incident was selected) and then alphabetically by the resource name.

Resources may have one of three possible statuses:

- *Available*. A resource is available if it is not currently being used to respond to an incident. New resources entered into the system are available by default. It is also assumed that available resources are located at the coordinates listed as the home location.

- *In Use*. A resource is in use if it has been deployed to respond to an incident. A given resource cannot be used to respond to multiple incidences at the same time—i.e., a resource must return to the available status before it can be in use again.

- *In Repair*. Repairs can be scheduled by the owner.

The most important feature of ERMS is the ability to request needed resources from other users to respond to an incident. A resource can be requested directly from the search results screen only when the user selected an incident on the search criteria form. Resource requests are always made within the context of an incident—i.e., you cannot request a resource without first selecting an incident. Both available and in use resources may be requested because a request does not imply immediate deployment of the resource. In fact, multiple requests may queue up for the same resource. However, a resource may only be deployed to respond to one incident at a time. Resources currently being repaired can not be requested. When Resources are requested, an expected return date must be added, to give at least an estimated time for when the resource will be returned.

Further, the owner can schedule a repair on any resource. Repair requests should be of the form "for N days after return" for *In Use* resources and "for next N days" for *available* resources. The *Next Available* date should take into account the requested repair. How requests work during repair durations:

a. If the repair duration has already begun, the repair can not be cancelled and hence the resource can not be deployed/requested.
b. If the duration is yet to begin, the owner can accept requests for it but will have to explicitly cancel the repair request (on the resource status page).

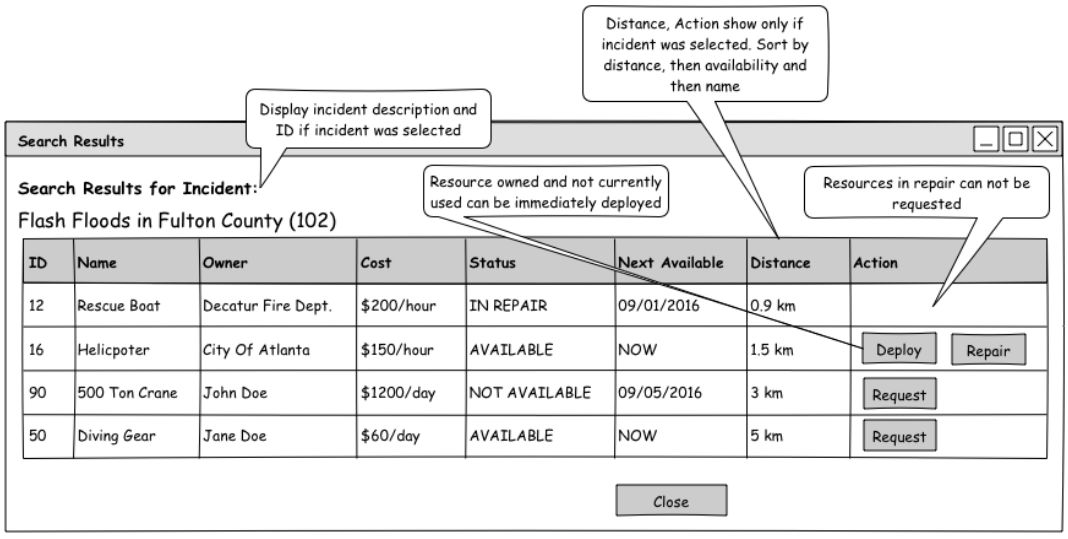


Figure 6 - Search Results

Resource Status

The resource status screen (linked from the main menu) shows the status of resources that are in use or have been requested. As shown in Figure 7, the resource status screen is divided into four sections.

The top section lists all resources that are currently in use responding to any incidents owned by the current user. For each resource, the **ID**, **name**, **incident responding to**, and **owner** are listed. The user also has the ability to return the resource to available **status** by clicking the *return* button for that resource. Resources must be returned to the available status before they can be deployed again. Once a resource has been returned back to available status, the system should prevent the same resource from being requested again for the same incident. However, the returned resource may be requested to respond to other incidents.

The middle section lists resource requests that have been sent by the current user (to another user) but have not yet been responded to. For each resource, list the **ID**, **name**, related **incident**, and **owner**. Note that the incidents listed in this box are all owned by the current user and the resource owners listed in this box are users other than the current user (because you cannot request resources from yourself). The system should allow the user to *cancel* any pending resource requests. In this case, the request will disappear from the other user's list of received requests.

The bottom box on the resource status screen lists resource requests received by the current user that are awaiting the user's response. For each resource, list the **ID**, **name**, related **incident**, and the **requesting user**. From here, the current user has two options: *deploy* the resource thereby accepting the other user's request or *reject* the request altogether. If the resource is deployed, then its **status** should be set to in use. If the request is rejected, then the pending request should be removed from the current user and requesting user's resource status screens. It is possible that a rejected resource request might appear again if the requesting user performs another search and requests the resource again. Note that the *deploy* button for the last resource (with ID 6) in Figure 7 has been disabled. This is because resource 18, which is requested to respond to Midtown Building Collapse, is currently in use responding to North GA Landslide (see the first row of Resource in Use). After resource 6 has been returned (after the user clicks the *return* button), it will be available for deployment to respond to Midtown Building Collapse. In no circumstances should the system allow a resource that is currently in use/repair be deployed to respond to another incident.

A note regarding data modeling of resource requests: Even though a user cannot formally request her own resources (i.e., they are immediately deployed), it is permissible behind the scenes to perform a "hidden request" followed by an immediate deployment. Doing so allows for the resource request and deployment process to be identical for resources owned by the current user and by other users.

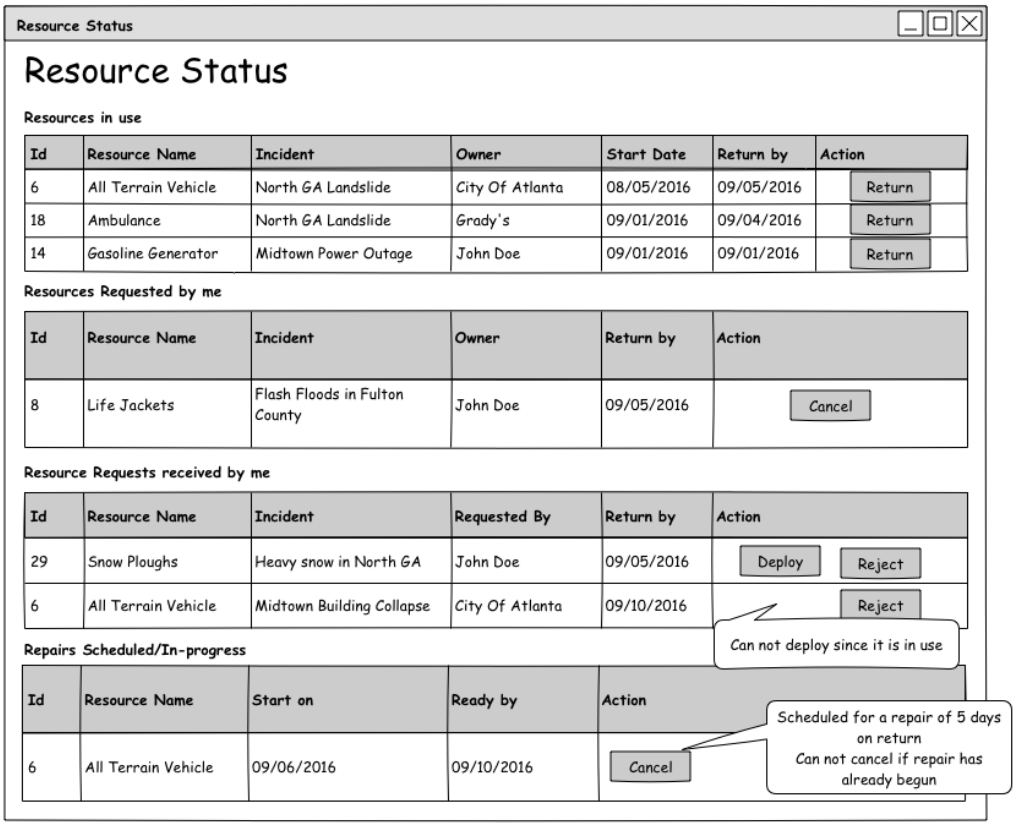


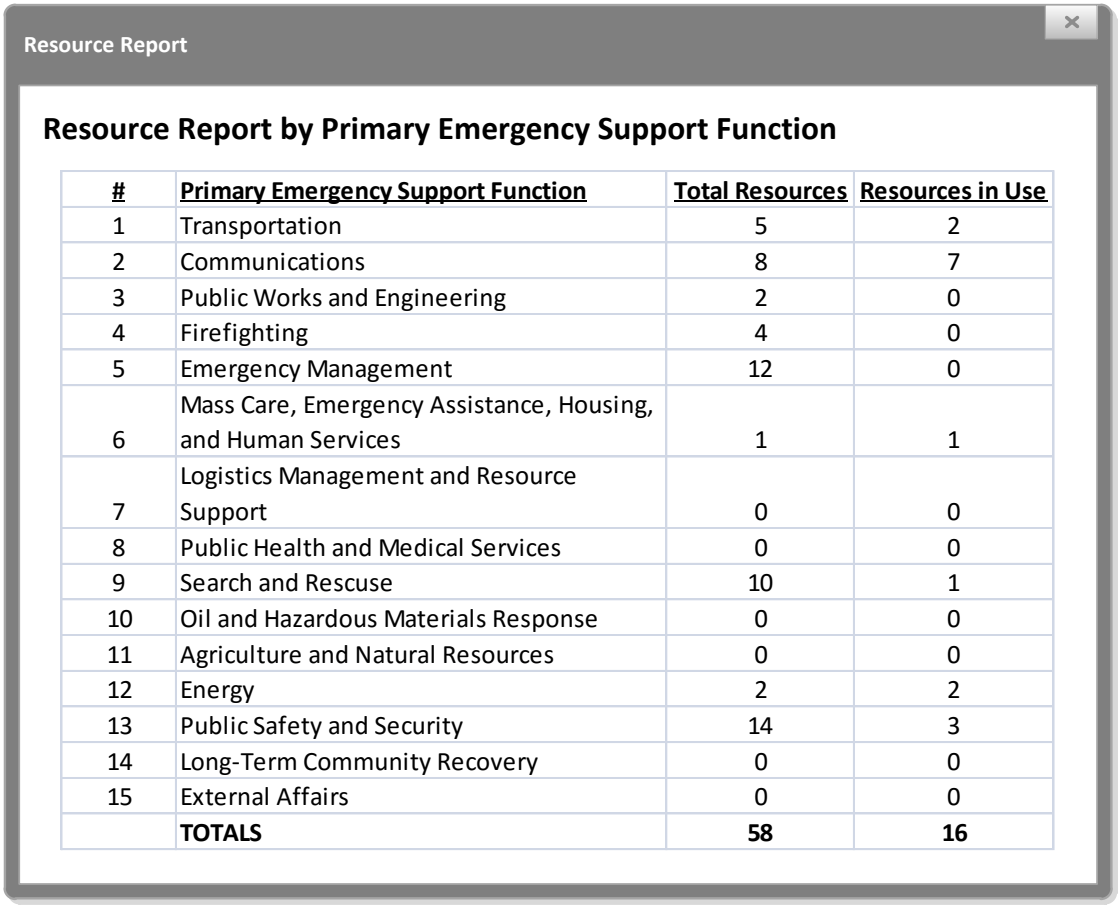
Figure 7 - Resource Status

Resource Report

From the main menu, the user has the option of running a report that summarizes the number of resources the current user owns grouped by the primary Emergency Support Function (ESF). An example report is shown in Figure 8. The report has four columns including: the ESF number, the ESF description, the total number of resources the user owns for that ESF, and the number of resources currently in use under that ESF.

- This report should only consider the primary ESF for each resource and ignore the additional ESFs field.
- Only resources owned by the current user should be counted for the Total Resources column.
- The Resources in Use column counts the total number of resources with the given ESF that are currently in use (i.e., deployed).
- All ESFs should be shown, even if the user owns no resources for that ESF.

- The last row shows the total number of resources owned by the user and the total number of resources currently in use.



#	Primary Emergency Support Function	Total Resources	Resources in Use
1	Transportation	5	2
2	Communications	8	7
3	Public Works and Engineering	2	0
4	Firefighting	4	0
5	Emergency Management	12	0
6	Mass Care, Emergency Assistance, Housing, and Human Services	1	1
7	Logistics Management and Resource Support	0	0
8	Public Health and Medical Services	0	0
9	Search and Rescuse	10	1
10	Oil and Hazardous Materials Response	0	0
11	Agriculture and Natural Resources	0	0
12	Energy	2	2
13	Public Safety and Security	14	3
14	Long-Term Community Recovery	0	0
15	External Affairs	0	0
	TOTALS	**58**	**16**

Figure 8 - Resource Report