Using GIS to Make an Electrical Engineering Firm's Project Data More Accessible

Karen E. Rauschert

The Pennsylvania State University

Index

Using GIS to Make an Electrical Engineering Firm's Project Data More Accessible

Background Information

I work as a drafting and GIS manager for an electrical engineering firm, Engineered Solutions Group (ESG), which has grown significantly since I first started in 2003. The firm is comprised of six companies that specialize in consulting, construction, maintenance, testing, and commissioning. Projects are mainly electrical in nature including power plants, substations, transmission, distribution, hydro energy, supervisory control and data acquisition (SCADA), right-of-way, studies, and maintenance as well as some mechanical and civil work. Our main office is located in Anchorage, AK and remote offices are located in Alaska, Washington, and Kansas. In 2006 the total number of employees on the payroll was 101 which doubled to 202 in 2009. As the company has grown it has become evident that accessing data from previous projects is frequently a difficult and time-consuming task due to how the data is organized in the current system.

The current system for storing project data consists of a job list spreadsheet, five servers, and a multitude of other sources. The job list spreadsheet is an Excel document that is updated for every new job that is added and contains basic job information such as job number, client, contracting company, job name, and project manager. Information could be accessed from this spreadsheet more easily if every new job occupied a single row but many jobs occupy multiple rows to house the tasks for the job. Organizing the data in this fashion prevents users from quickly compiling data through sorts and counts.

There are five main servers that contain project data including servers for proposals, current jobs, archived jobs, project photos, and right-of-way information for permitting. The proposals server contains information acquired prior to a project obtaining a job number and is organized by contracting company, year, and then client. Once a job number is assigned it is not uncommon for the information to be copied to the current jobs server. Files are organized by client and then job number but users must have a license to access the files as well as set up a client work area and populate files to search, view, or edit them because revision control software (SOS) is used. No one computer has all client work areas created and files populated and space requirements prevent this so it is not possible to do a single search of everything on the current jobs server. Files exist on the current jobs server until they are moved to the archived jobs server which may be long after the job has been closed. The archived jobs server is also organized by client and then job number and contains files moved from the current jobs, project photos, and right-of-way servers. The project photos and right-of-way servers are also organized by client and then job number and are separate from the current jobs server because they do not require revision control software. Project photos are organized in dated directories but frequently have no additional labeling or descriptive file names which makes searching difficult.

Other sources where job data may be stored include Google Drive, Dropbox, and other company servers (such as one for SCADA). Most of these sources are not accessible to most employees and the data will likely not be archived if it is not added to one of the main servers. Also, a lot of project information is still obtained through word of mouth but as the company grows it is not uncommon for employees of one department to not know employees of another department. As individuals leave the company this information is frequently lost.

Problems with the current system include access to data, accuracy of data, and usefulness of data. Accessing the data may require the user to look in multiple locations and it is common to have duplicate and possibly conflicting information throughout the system. Users must know the job number and client to access most of the data. No one regularly monitors the data to ensure accuracy so it is not uncommon to find inconsistencies in file and project names and missing data. Some data has no standard location in the current system so created documents such as lessons learned are frequently not stored with the project data. I frequently need to access data for previous projects that I have worked on which typically involves opening multiple windows for various clients on both the current jobs and archived jobs servers until I locate the desired information. The inability to easily answer simple questions such as “What jobs have we completed in a specified location,” “What jobs have we used this equipment on before,” or “What types of projects has the company done” indicates that accessing data in the current system is failing to meet the needs of employees in a timely and cost-effective manner. This project is my attempt to combat some of the problems with the current system.

Literature Review

I conducted a literature review of multiple websites to help determine the design for the new site and to determine what approaches others have taken in the past. The sample sites could loosely be broken down into two broad categories – individual feature maps and project maps. An example of an individual feature map for a transmission line would be one where you could click on the individual poles and obtain data specific to that pole (Bonneville Power Administration, 2015) while a transmission project map would simply allow you to click on the entire transmission line and obtain project level data (EIA, 2015). A project map is what I was interested in creating, with the ability to input data for individual poles but not clickable features for individual poles.

There was also a multitude of ways the sample sites displayed both map and non-map data. Some sites used points and lines to display map data (WFRC, 2015) while others used polygons or varied based on the zoom level (Bonneville Power Administration, 2015). For my site I believed points for single locations like substations and power plants, lines for distribution and transmission lines, and polygons for study areas were sufficient. The sample sites also displayed non-map data in various ways including clickable entities (most sites), drop-down menus (Synbio Consulting, 2015), check boxes (NYCEDC, 2015), radio buttons (Pennsylvania Department of Transportation, 2015), and clickable lists (Montgomery County, MD., 2015). It was clear that some sites were more effective than others with keeping the map as the main focus for the site and allowing the user to easily display the data which I was hoping to achieve with my own site.

I envisioned a map with a combination of filters for displaying data. When the user hovered over a feature it would display the facility name. When a feature was clicked it would display a clickable list of all of the projects that were completed at that location and allow the user to get project-specific information by clicking on the project link. Drop-down menus would allow quick access to projects associated with specified fields such as location, client, project manager, or job number. Users would have the ability to search the site to quickly locate information. I also wanted to implement a clickable project list that would list all of the displayed projects. Many sites gave the user the choice of multiple base maps which I saw value in. The vision for my site was not based on a single example site but a compilation of components from multiple sites.

Application Development

In order to combat some of the problems with the current data storage system I developed a structured means of inputting and storing relevant project information that is robust, searchable, and reproducible. I accomplished this through the use of five specific aims – identifying relevant project information, developing a procedure to normalize data, developing and testing a relational database and testing the dataset, developing a system for data entry, and designing a user interface.

Relevant Project Information

Determining what data to collect based on its usefulness and availability was an important first step. ESG recently implemented the use of project creation and project closure forms to capture specific information from project managers when jobs were opened or closed (Appendix A). These forms became a valuable source of information and defined what basic project data would be easily obtainable. Additional data deemed essential or desirable was also included in relevant project information which was broken down into the following categories: projects, people, companies, dates, locations, major components, and documents. Contents of each category are listed below.

<u>Projects</u>

- Job Number
- Job Name
- Contract Type
- State Job (Y/N)
- Federal Job (Y/N)
- Type of Work
- Original Budget
- Final Budget
- Explanation of Difference in Original and Final Budgets
- Completion on Schedule (Y/N), Explanation if Not
- Deliverables
- Project Description (at Closure)
- Work Description (at Creation)
- Additional Information (at Creation)

<u>People</u>

- Project Manager
- Client Contact
    - Contact Email
    - Contact Phone

<u>Companies</u>

- Client
- Prime Contractor
- ESG Company
- Internal Subcontractors

<u>Dates</u>

- Event
- Date

<u>Locations</u>

- Shapefile (point, polyline, or polygon)
- Location Name
- Address
- City
- State
- Zip
- Country
- County
- Description

<u>Major Components</u>

- Component Name
- Manufacturer (If Applicable)
- Description
- Lessons Learned
- Link to Document

<u>Documents</u>

- File Title
- Link
- Associated Component (If Applicable)
- Description

There is a great deal of variability among projects so some projects will have very little data to include while others will need to be narrowed down to include just the appropriate data for this application. Projects will require a geographic tie and must be non-active so only timeless information is entered into the system. Locations such as power plants and substations would be displayed as points, transmission and distribution lines would be displayed as lines, and larger areas such as study areas would be displayed as polygons. The points, lines, and polygons will be for individual facilities (not projects) to avoid overlapping records and to create a geographic tie between projects that occur at the same location. Major components can be used to capture some of the variability among projects and may include items such as equipment, poles/structures, parcels, conductor, foundations, communications, subcontractors, existing utilities, or special designs such as avalanche design. Examples of documents that might be worth incorporating include reports, photos, drawings, lessons learned, proposals, material cut sheets, and plats.

Data Normalization/Data Integrity

After establishing relevant project information, I developed a database design that incorporated both normalization and data integrity. The contact and location tables were created as separate tables from the main project table and do not use the job number to tie them to the project table in order to eliminate the amount of duplicate information being stored (many client contacts are the same for multiple jobs and many jobs take place in the same location). Multiple jobs might also use the same components; however I decided not to combine components from multiple jobs because part numbers change over time and records may still contain project specific information such as lessons learned. The creation of separate tables for component, dates, link, project location, and type of work allow multiple records to be captured for a single job without the need for additional columns.

I constructed a data dictionary to define the information being collected and relationships among the data (Appendix B). All of the data for projects, people, and companies will likely come directly from the job creation and closure forms which each have instructions defining what should be entered in each of the fields. In the future, data collection for the job creation and closure forms could be improved by switching to an online form format that constrained additional data and would necessitate all required fields be completed before the form could be submitted. Right now data verification is mostly done manually and individuals reject forms that are not completed. The data for dates, locations, components, and documents will be pulled from various sources so defining these fields is even more important. In the future it would be helpful to collect some of this additional information at the time of job creation and/or closure such as specific project locations (currently only city, county, and state are collected).

Two additional timestamp columns were added to each of the tables to record when each record was initially created and last modified. This information can be used for tracking the data entry process and may prove helpful for accessing the accuracy of the data.

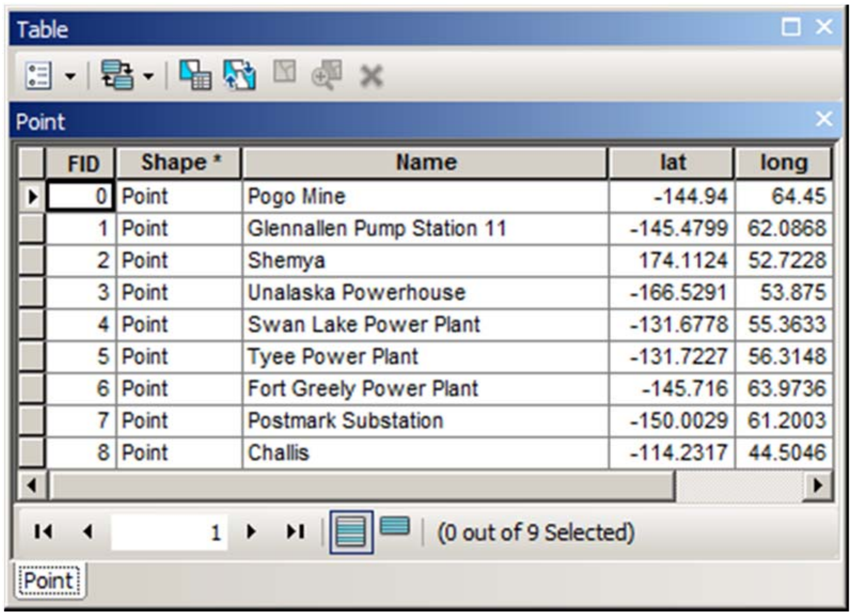
Relational Database

MySQL was used for the relational database which contains eight main tables including project, contact, dates, location, project location, component, link, and type of work. Most of the tables (all of them except contact and location) contain the project job number which ties them together. The contact table is tied to the project table using the contact’s first and last name fields (a separate table was created to avoid duplication of the associated contact data). The location table is tied to the project location table using the location name field. Separating these tables eliminates the duplication of location data while allowing multiple locations to be associated with a single project when applicable. The dates, project location, component, link, and type of work tables were all created as separate tables from the project table to allow for the variability among projects and incorporate the flexibility of adding zero (or one for project location) to multiple records for each project. I also created multiple import tables to temporarily house data and simplify the data entry process.

Data Entry

The geometry data (points, polylines, and polygons) is stored in shapefiles along with location names as shown in Figure 1, while the remaining data is stored in database tables.

Location names provide the link between the shapefiles and database tables. Geometry data is entered manually using ArcMap.

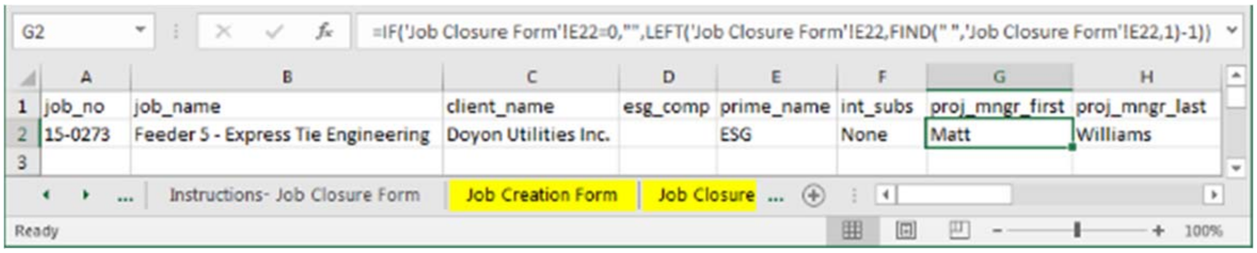


FID	Shape *	Name	lat	long
0	Point	Pogo Mine	-144.94	64.45
1	Point	Glennallen Pump Station 11	-145.4799	62.0868
2	Point	Shemya	174.1124	52.7228
3	Point	Unalaska Powerhouse	-166.5291	53.875
4	Point	Swan Lake Power Plant	-131.6778	55.3633
5	Point	Tyee Power Plant	-131.7227	56.3148
6	Point	Fort Greely Power Plant	-145.716	63.9736
7	Point	Postmark Substation	-150.0029	61.2003
8	Point	Challis	-114.2317	44.5046

(0 out of 9 Selected)

Figure 1. Point shapefile attribute table

A significant amount of the data is pulled directly from the job creation and closure forms which are in Excel format. A new sheet is created within the Excel forms file and formulas and functions are used to populate the included fields as shown in Figure 2. This example shows the project manager's name being retrieved from the job closure form and split into first and last name.

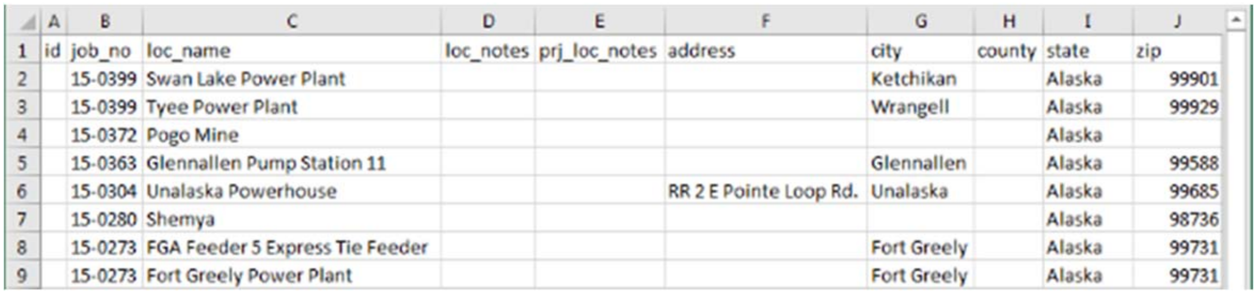


=IF('Job Closure Form'!E22=0,"",LEFT('Job Closure Form'!E22,FIND(" ",'Job Closure Form'!E22,1)-1))

	A	B	C	D	E	F	G	H
1	job_no	job_name	client_name	esg_comp	prime_name	int_subs	proj_mngr_first	proj_mngr_last
2	15-0273	Feeder 5 - Express Tie Engineering	Doyon Utilities Inc.		ESG	None	Matt	Williams
3								

Figure 2. Implementing Excel formulas/functions to pull data from job creation/closure forms

Data that is not provided on the job creation and closure forms such as location data must be entered manually as shown in Figure 3.



	A	B	C	D	E	F	G	H	I	J
1	id	job_no	loc_name	loc_notes	prj_loc_notes	address	city	county	state	zip
2		15-0399	Swan Lake Power Plant				Ketchikan		Alaska	99901
3		15-0399	Tyee Power Plant				Wrangell		Alaska	99929
4		15-0372	Pogo Mine						Alaska	
5		15-0363	Glennallen Pump Station 11				Glennallen		Alaska	99588
6		15-0304	Unalaska Powerhouse			RR 2 E Pointe Loop Rd.	Unalaska		Alaska	99685
7		15-0280	Shemya						Alaska	98736
8		15-0273	FGA Feeder 5 Express Tie Feeder				Fort Greely		Alaska	99731
9		15-0273	Fort Greely Power Plant				Fort Greely		Alaska	99731

Figure 3. Manual entry of location data

Once the data is entered in Excel, it is saved as a comma-separated values (csv) file and loaded into the MySQL database using the MySQL console. The data is initially imported into temporary import tables and then the applicable tables are populated from the data in the import tables. Data integrity and quality are checked throughout the data entry process. Starting with the most recent non-active projects and working backwards would be a good approach for initially getting all of the desired projects into the system. Older projects will not have job creation and closure forms to pull data from.

User Interface

When a user first accesses the site the user is presented with a map-based web application built using the Google Maps Application Programming Interface (API), an empty table, and two buttons with options to “Show All Jobs” or “Select Job Criteria” as shown in Figure 4. The “Show All Jobs” option displays all of the point, polyline, and polygon location data currently saved within the system (in shapefile format) and tabular data for job number, job name, location name, client, project manager, and state (saved in MySQL).

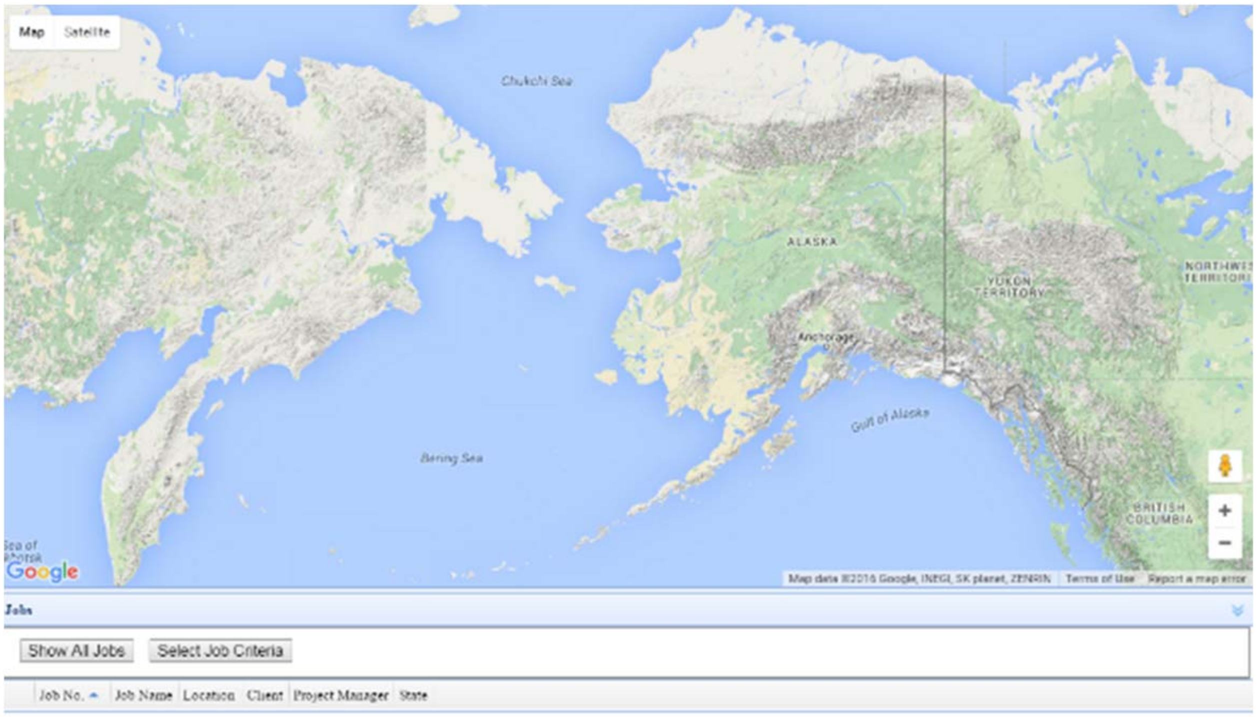
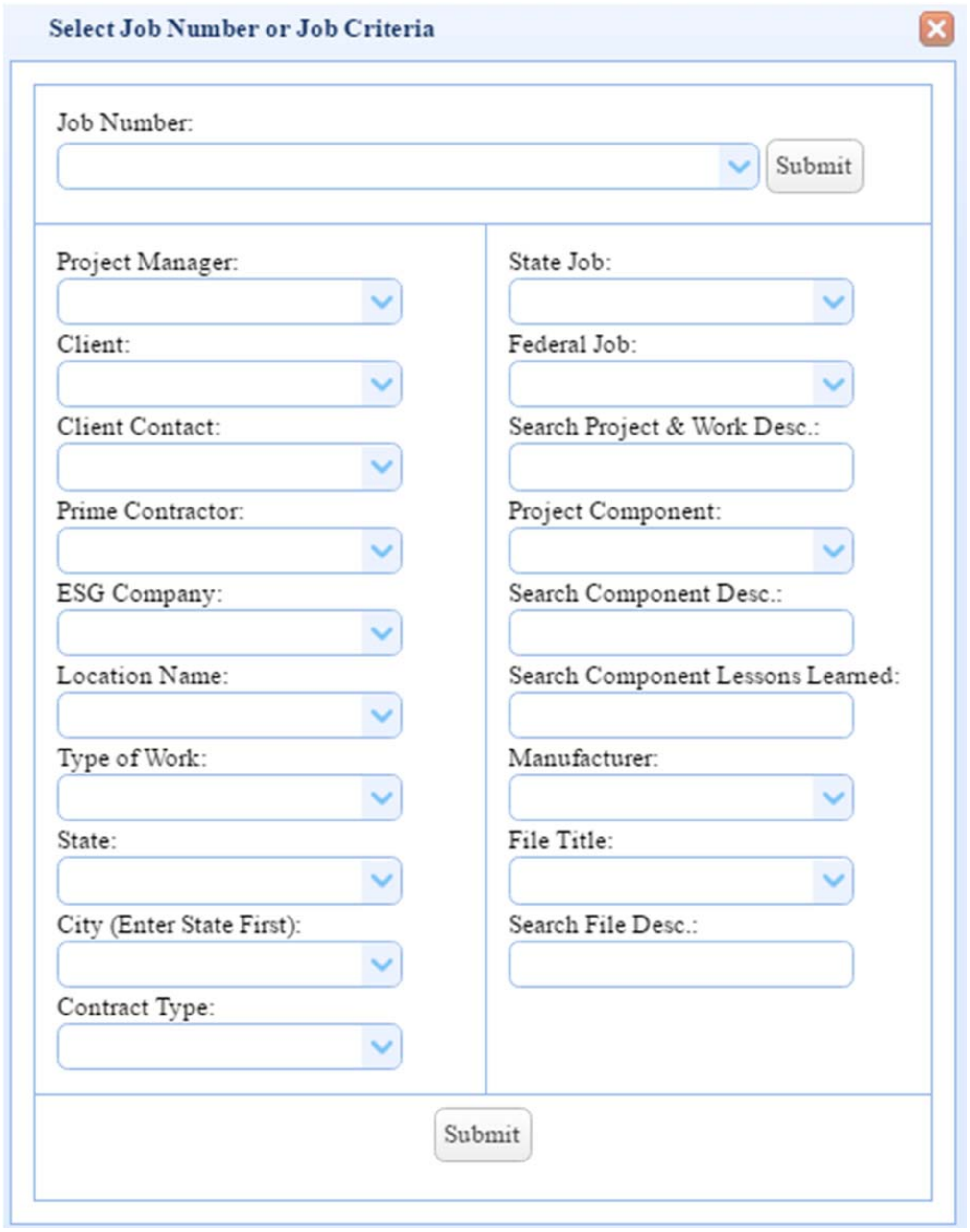


Figure 4. Initial default page for user interface

The “Select Job Criteria” option displays a dialog with combo boxes and text boxes as shown in Figure 5. The user can choose a single job based on job number or choose jobs that meet the specified criteria for project manager, client, client contact, prime contractor, ESG company, location name, type of work, state, city, contract type, state job (Yes/No), federal job (Yes/No), project component, manufacturer, file title, and text searches for project and work description, component description, component lessons learned, and file description. Each combo box is populated with a list of values from the database and the user has the option to type his/her selection for quicker completion. The city combo box list is not populated until a state is selected.



Select Job Number or Job Criteria

Job Number:

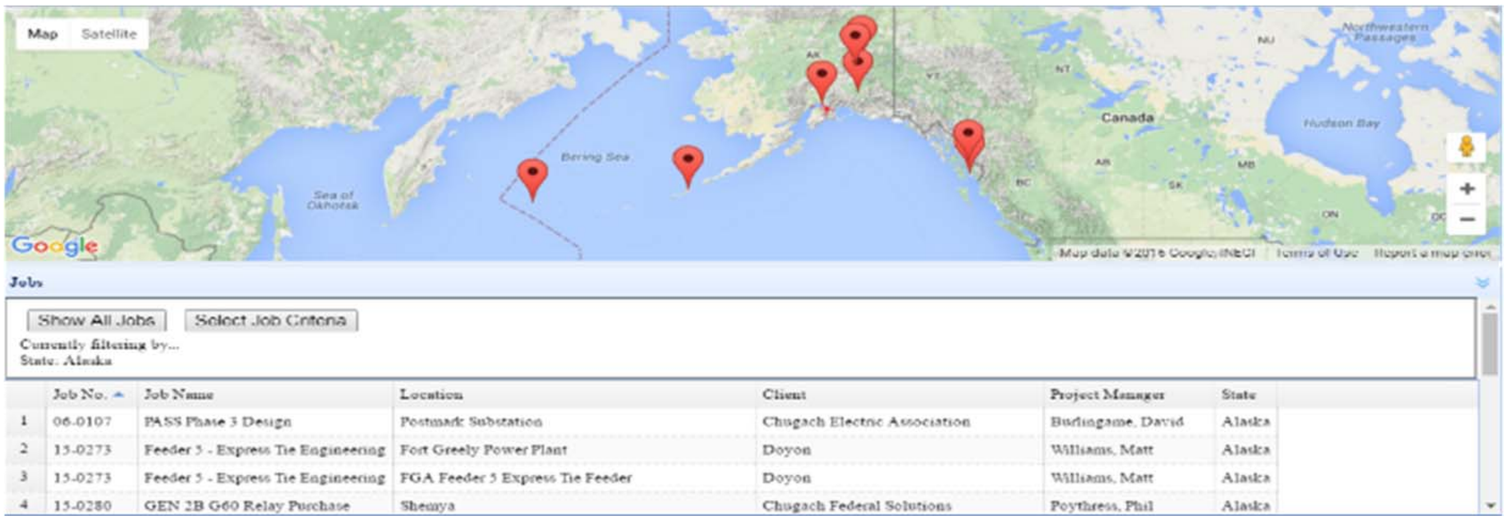
Submit

Project Manager:	State Job:
Client:	Federal Job:
Client Contact:	Search Project & Work Desc.:
Prime Contractor:	Project Component:
ESG Company:	Search Component Desc.:
Location Name:	Search Component Lessons Learned:
Type of Work:	Manufacturer:
State:	File Title:
City (Enter State First):	Search File Desc.:
Contract Type:	

Submit

Figure 5. Dialog box for selecting job criteria

Once a user completes his/her selection and clicks the submit button the results are displayed as shown in Figure 6. In addition to the map showing the points, polylines, and/or polygons and the populated table, just above the table the site also displays what job filters were selected.



Jobs

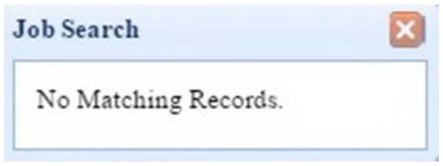
Show All Jobs | Select Job Criteria

Currently filtering by...
State: Alaska

	Job No.	Job Name	Location	Client	Project Manager	State
1	06-0107	PASS Phase 3 Design	Postmark Substation	Chugach Electric Association	Burlingame, David	Alaska
2	15-0273	Feeder 5 - Express Tie Engineering	Fort Greely Power Plant	Doyon	Williams, Matt	Alaska
3	15-0273	Feeder 5 - Express Tie Engineering	FGA Feeder 5 Express Tie Feeder	Doyon	Williams, Matt	Alaska
4	15-0280	GEN 2B G60 Relay Purchase	Shemya	Chugach Federal Solutions	Poythress, Phil	Alaska

Figure 6. Selection using job criteria

If no jobs match the selected criteria a dialog is displayed with the text “No Matching Records.” as shown in Figure 7. The user must close this dialog before proceeding.


Job Search

No Matching Records.

Figure 7. Dialog for no matching records for job criteria selection

When a user hovers over a location, the location name is displayed as shown in Figure 8. When a user clicks a location or a row in the table, job specific information is displayed on an info window which includes job number, client, project manager, state, and job name as well as a button for displaying additional project data. If there is data for more than one job for a single location, then multiple jobs are displayed in the info window as shown in Figure 9.

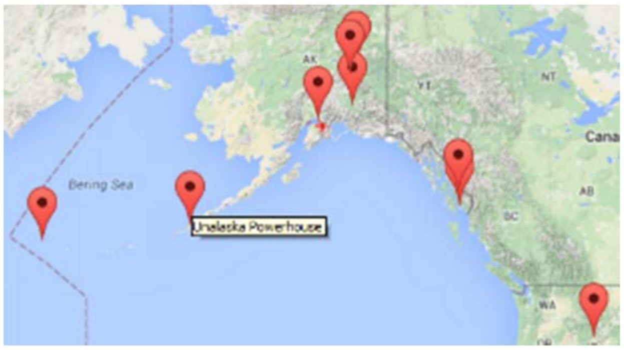
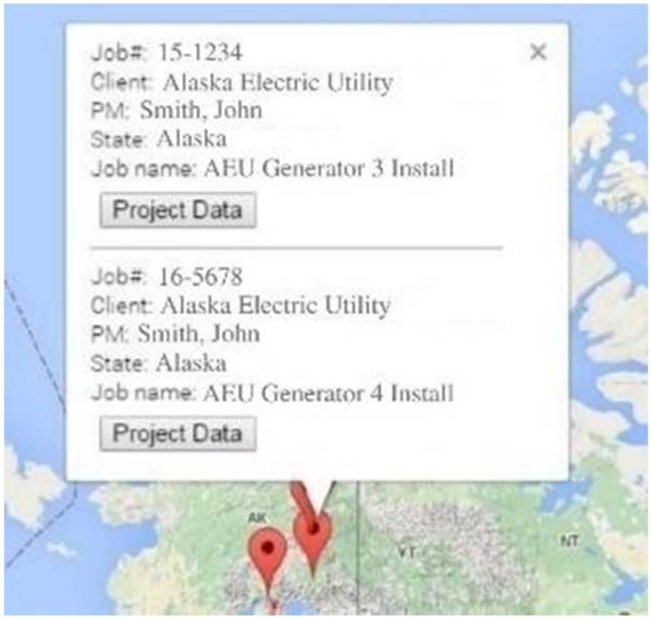


Figure 8. Display of location name when hovered over location

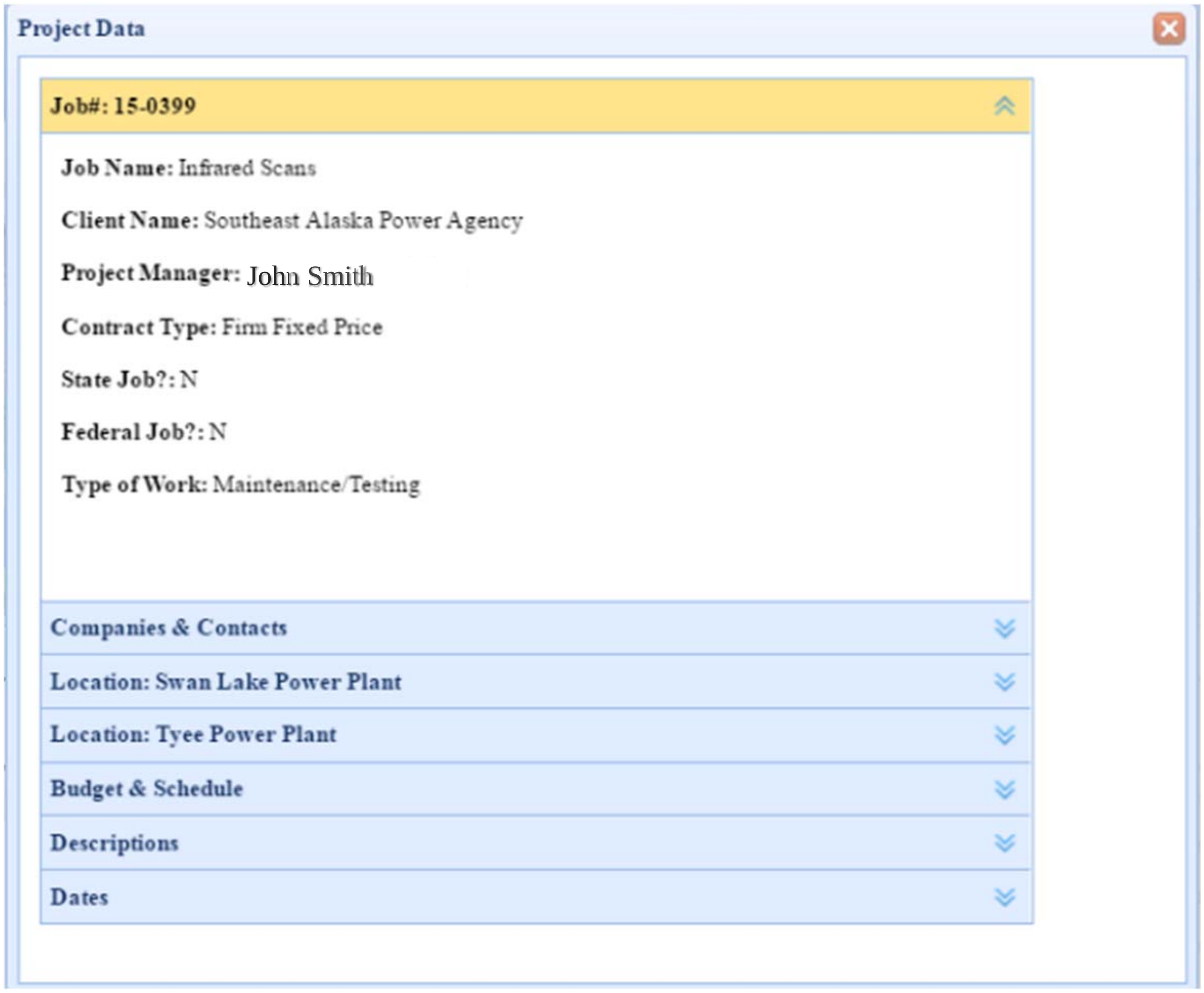


Job#: 15-1234
Client: Alaska Electric Utility
PM: Smith, John
State: Alaska
Job name: AEU Generator 3 Install
Project Data

Job#: 16-5678
Client: Alaska Electric Utility
PM: Smith, John
State: Alaska
Job name: AEU Generator 4 Install
Project Data

Figure 9. Info window of job specific information when location is clicked

When a user clicks on a "Project Data" button an accordion style dialog box appears displaying even more job specific information as shown in Figure 10. There are accordion tabs for general project information, companies and contacts, location(s), budget and schedule, descriptions, dates, components, and links. It is designed to only show the tabs and information on the tabs that are available (not null) for the specified job so what is displayed will vary from job to job.


Project Data

Job#: 15-0399

Job Name: Infrared Scans

Client Name: Southeast Alaska Power Agency

Project Manager: John Smith

Contract Type: Firm Fixed Price

State Job?: N

Federal Job?: N

Type of Work: Maintenance/Testing

Companies & Contacts

Location: Swan Lake Power Plant

Location: Tyee Power Plant

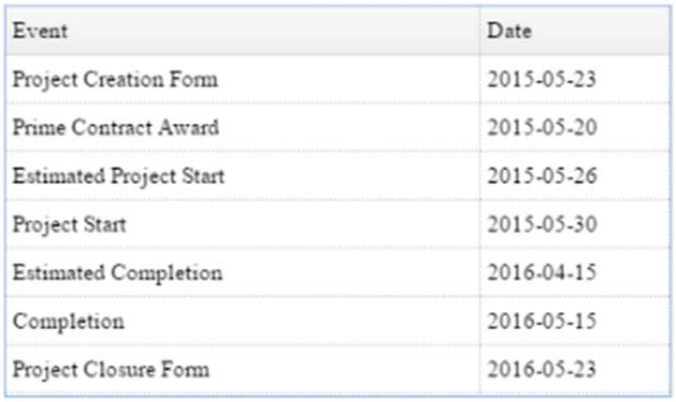
Budget & Schedule

Descriptions

Dates

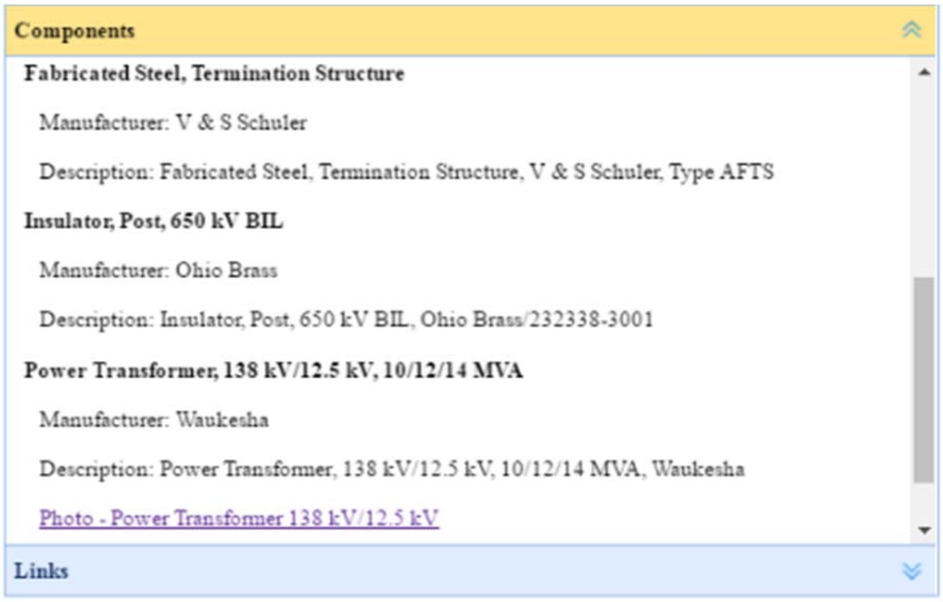
Figure 10. Accordion style dialog of job information shown when project data button is clicked

The general project information may include job number, job name, client name, project manager, contract type, state job (Y/N), federal job (Y/N), and type of work. The companies and contacts tab may include prime contractor, ESG company, project manager, internal subs, client name, client contact, contact email, and contact phone (up to two phone numbers). A location tab will be displayed for all project locations and may include location name, address, city, state, zip, country, county, and description. The budget and schedule tab may include original budget, final budget, explanation of difference in budgets, whether or not the job was completed on schedule, and an explanation of why not. The descriptions tab may include deliverables, project description at job closure, work description at job creation, and additional information at job creation. If dates were entered for a job they will be displayed in a table including events and dates as shown in Figure 11. The components tab may include the component name, manufacturer, description, lessons learned, and links to files as shown in Figure 12. The links tab may include file title, link, associated component, and description.



Event	Date
Project Creation Form	2015-05-23
Prime Contract Award	2015-05-20
Estimated Project Start	2015-05-26
Project Start	2015-05-30
Estimated Completion	2016-04-15
Completion	2016-05-15
Project Closure Form	2016-05-23

Figure 11. Dates table in accordion style dialog



Components

Fabricated Steel, Termination Structure

Manufacturer: V & S Schuler

Description: Fabricated Steel, Termination Structure, V & S Schuler, Type AFTS

Insulator, Post, 650 kV BIL

Manufacturer: Ohio Brass

Description: Insulator, Post, 650 kV BIL, Ohio Brass/232338-3001

Power Transformer, 138 kV/12.5 kV, 10/12/14 MVA

Manufacturer: Waukesha

Description: Power Transformer, 138 kV/12.5 kV, 10/12/14 MVA, Waukesha

Photo - Power Transformer 138 kV/12.5 kV

Links

Figure 12. Components tab in accordion style dialog

Code Overview

Early on I decided against using dedicated web mapping software such as ArcGIS Server to build my site and instead took advantage of software, much of it free and open-source, that would not incur new costs for my company. This resulted in a clunkier, more piecemeal assembly but I was able to accomplish what I set out to do without any additional software expenses. The software products I used include MySQL for the database, PHP for the server scripting language, Apache for the web server software, and Windows 7 for the operating system (the site will be run on a Linux server once implemented). Shapefiles were used for the storage of geometry data in order to simplify data entry. The Google Maps API was used for the mapping portion of the site. jQuery (a JavaScript library) was also used for some of the coding and jQuery EasyUI was used for some of the graphical user interface (GUI) elements. The site is intended to be an intranet site that is only accessible to ESG employees.

The code consists of the main html file (Appendix C), which calls on 18 php files – one for getting location and table data for selected locations (Appendix D), one for getting project data for the accordion dialog (Appendix E), and 16 for populating the combo box lists (Appendix F). Roughly the first 379 lines of the html file are written in HTML while the remainder of the file is primarily JavaScript. The HTML portion of the code mainly defines the layout and styles for the site and provides links to required libraries for jQuery, jQuery EasyUI, Google Maps API, and a marker label used when hovering over polylines and polygons. All of the libraries are stored locally with the exception of the Google Maps API (Google's Terms of Service prohibit it) but the jQuery library is only accessed locally if the website cannot be accessed.

The JavaScript portion of the code starts by declaring global variables (lines 380-390). Combo box values are validated in lines 392-458 and the bulk of the rest of the file (lines 460-1081) consists of 27 functions.

When the site first loads the initMap function is run which creates the map and loads in the shp and dbf files for the shapefiles. The initMap function calls on four other functions (shpLoad, shpLoadError, dbfLoad, and dbfLoadError) for loading the shapefiles and two of these functions call on the reorder function to put the files back in the correct sequence. The shpLoad and dbfLoad functions were borrowed from the Penn State GIS Mashups course (GEOG 863) and were originally adapted from a Google example. The data loading for the shp and dbf files is done asynchronously so the reorder function ensures that the correct shp file (geometry data) is matched up with the correct dbf file (tabular data).

Once the map is loaded the user can either select the “Show All Jobs” button or the “Select Job Criteria” button. If the “Show All Jobs” button is selected the clearFilterList and render functions are run. The clearFilterList function will clear the filter text if job criteria were previously selected. The render function initiates the showLoading function which lets the user know the requested map is still loading and the clearMap function, which clears the info windows, table, and locations for previous selections. The get_location.php script is called on to run a query for all jobs and returns the results. The info window is defined and the points, polylines, polygons, and table are added to the map using the addToSidebar, createMarkers, createLines, and createPolys functions. Additional functions are needed for the creation of the polylines and polygons (pathToArray, makePolyline, and makePolygon) and markers are also added at this time for location name labels for polylines and polygons. The addToSidebar function sets up a listener for clicks on the table and opens the appropriate map info window in the event such a click happens. Likewise, createMarkers, createLines, and createPolys set up listeners for clicks on the map features themselves and open the appropriate info window. The getLineMidpoint and getBoundsForPoly functions are called on to determine info window positions for polylines and polygons. Finally, the hideLoading function is called to hide the notification for the user that the map is loading.

If the user displays an info window (by clicking a location or a table row) he/she has the option of clicking the “Project Data” button. When clicked the get_job_info.php script runs a query using the job number and returns the job specific data. The openDialog function is also called on to open the Project Data accordion dialog.

If the “Select Job Criteria” button is selected, the “Select Job Number or Job Criteria” dialog is opened. All but the city combo box lists are populated in the HTML portion of the code by calling on the 15 php list scripts. The city combo box list is populated when a state is selected and cleared if the dialog box is closed. Clicking on a submit button (there are two in the dialog) initiates the process function. The process function gets the user-entered values and passes them to the render function. The project manager and client contact values must both be split into two variables because first and last names are stored in separate fields in the database. The filter list is also created in the process function for displaying the user’s selection. The field names are used for this but the underscores are replaced with spaces and the titleCase function is used to capitalize the first letter of each word in the field name. The clearLists function is called to clear all of the combo box and text box values. The render function goes through the same steps as above when the “Show All Jobs” button was selected except this time the query uses the user-entered values from the process function and compares them to the database values returning only the matching records. If there are no matching records the “No Matching Records.” dialog is opened.

Assessment of the New Site and Future Developments

It is a struggle to answer these simple questions with the current system: “What jobs have we completed in a specified location,” “What jobs have we used this equipment on before,” or “What types of projects has the company done.” To answer the first question with the current system, I would likely do a search of the job list spreadsheet for the location name or search under clients that might do work in the specified area. With the new site the locations are visible on the map or a user can search by location name, state, city, keyword in the project description, or job name. For tracking down a job using a specified piece of equipment in the current system it would likely require previous knowledge if the equipment name was not part of the job name. In the new site a user can search by component name, manufacturer, job name or do a keyword search of the project description or component description. For determining what types of projects have been completed in the current system, I would again likely rely heavily on the job names. The new site allows a user to search by job name, type of work, and keyword searches but also houses all of the data in a single location for easy access. While the current system requires accessing data from multiple locations and relies on descriptive job names and previous knowledge, the new site strives to bring the relevant information into a centralized location and provide many search options for quickly accessing the data. Before the app, I would have to look in multiple client directories on both the current jobs and archived jobs servers for the desired information. Using the app, I can find the information by going to a single location and searching by project manager, component name, keyword, or other searchable features.

In the future I would like to improve on both the data entry process and the user interface. It would be advantageous to have additional information collected at the time of job creation or closure from the project manager to decrease the time required for data collection. The additional information I would want to request from project managers includes project locations, lessons learned, significant project dates, major project components, and significant project documents. The inclusion of this information would give users a better idea of what the project involved and allow it to be collected from the individual overseeing the project upon completion of that project. In addition, creating online forms for job creation and closure could allow the data to be fed directly into the database making the data entry process much more automated.

When the site is made accessible to employees it will be useful to make improvements to the user interface based on user input. It is likely that different users (engineers, project managers, administrators, etc.) would be accessing the site for a multitude of reasons (searching data for a specific or similar job, proposal writing, getting an overview of jobs completed, etc.) and might find it useful to include additional information or make changes to how the data is presented. Requesting this information from users and implementing it when possible and appropriate could better suit the site to users' needs. Expanding the site to be accessible to current clients and potential future clients is another prospective future development that could likely be accomplished through minimal changes to the internal site.

References

Bonneville Power Administration. *BPA Transmission Lines.* Retrieved on March 22, 2015 from http://www.arcgis.com/home/webmap/viewer.html?webmap=262e5ef1de424730b096a0eed94035a5

EasyUI. (2016). *Welcome to jQuery EasyUI.* Retrieved from the EasyUI website on May 2016: http://www.jeasyui.com/index.php

EIA (U.S. Energy Information Administration). *U.S. Energy Mapping System.* Retrieved on March 22, 2015 from http://www.eia.gov/state/maps.cfm

Lerdorf, R., Tatroe, K., & MacIntyre, P. (2006). *Programming PHP.* Sebastopol, CA: O'Reilly Media, Inc.

Montgomery County, MD. *Development Projects in Montgomery County, MD.* Retrieved on March 22, 2015 from http://mcatlas.org/devfinder/

NYCEDC. *Interactive Map of Projects.* Retrieved on March 22, 2015 from http://www.nycedc.com/projects/projects-map

Pennsylvania Department of Transportation. *MPMS-IQ – Multi-modal Project MAPPING System Interactive Query*. Retrieved on March 22, 2015 from http://www.dot7.state.pa.us/MPMS_IQ/Mapping#

Petroutsos, E. (2014). *Google Maps: Power Tools for Maximizing the API.* New York, NY: McGraw Hill Education.

PHP Group, The. (2016). *PHP: Hypertext Preprocessor.* Retrieved from the PHP Group website on May 2016: http://php.net

Synbio Consulting. *iGEM Synthetic Biology Ecosystem.* Retrieved on March 22, 2015 from http://synbioconsulting.com/igem-synthetic-biology-map/

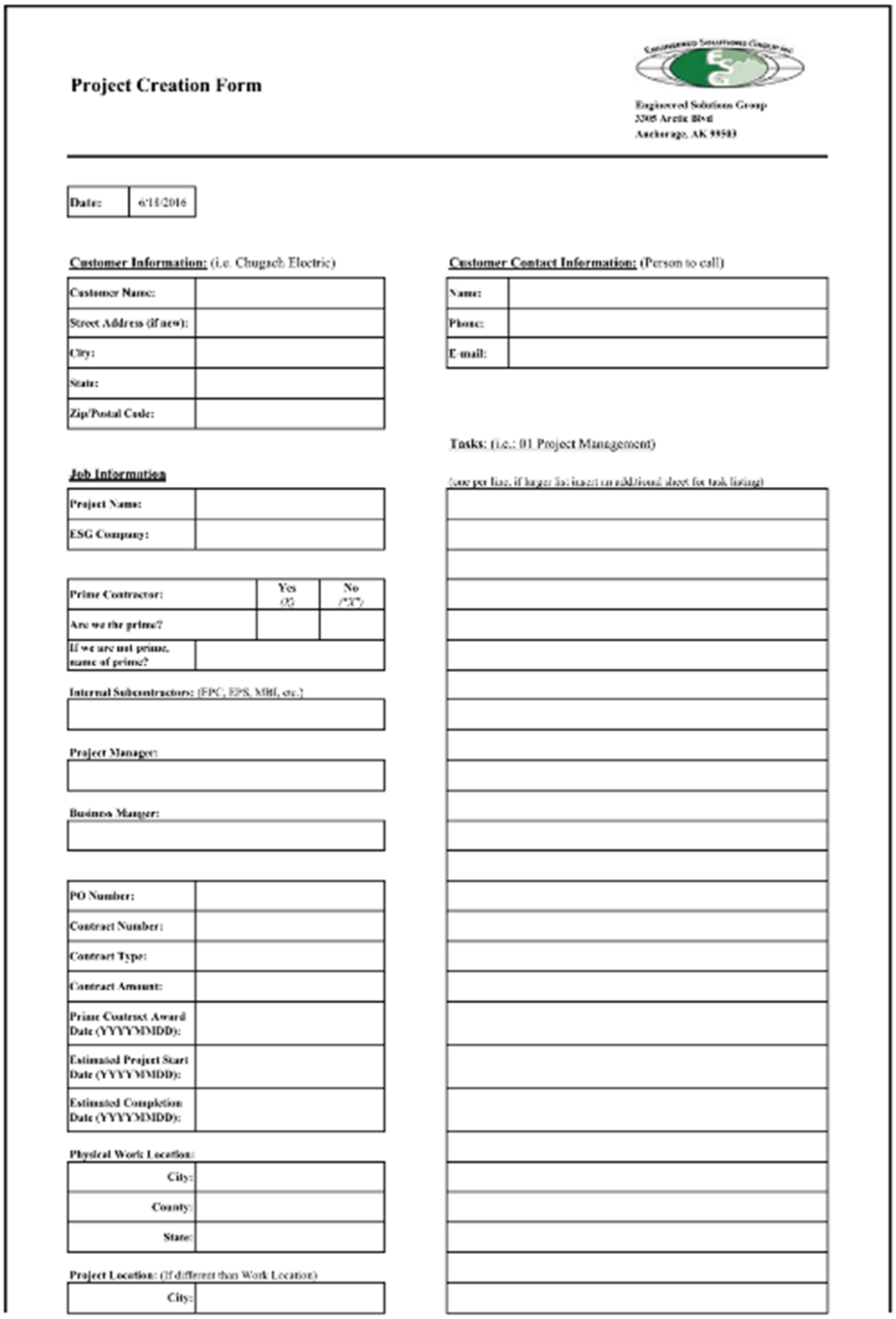
Welling, L., & Thomson, L. (2009). *PHP and MySQL Web Development*. Upper Saddle River, NJ: Addison-Wesley.

WFRC. *WFRC 2015-2040 Regional Transportation Plan.* Retrieved on March 22, 2015 from http://wfrcgis.maps.arcgis.com/apps/Viewer/index.html?appid=d4d106b5c4074233837e6fb5e9f4ee51

W3Schools. (2016). *jQuery Tutorial*. Retrieved from W3Schools website on May 2016: http://www.w3schools.com/jquery/default.asp

Appendix A

Project Creation Form & Project Closure Form



Project Creation Form

Engineered Solutions Group
3305 Arctic Blvd
Anchorage, AK 99503

Date:	6/11/2016

<u>Customer Information:</u> (i.e. Chugach Electric)

Customer Name:	
Street Address (if new):	
City:	
State:	
Zip/Postal Code:	

<u>Customer Contact Information:</u> (Person to call)

Name:	
Phone:	
E-mail:	

<u>Job Information</u>

Project Name:	
ESG Company:	

Prime Contractor:	**Yes** *(X)*	**No** *("X")*
Are we the prime?		
If we are not prime, name of prime?		

Internal Subcontractors: (EPC, EPS, MBI, etc.)

Project Manager:

Business Manger:

PO Number:	
Contract Number:	
Contract Type:	
Contract Amount:	
Prime Contract Award Date (YYYYMMDD):	
Estimated Project Start Date (YYYYMMDD):	
Estimated Completion Date (YYYYMMDD):	

Physical Work Location:

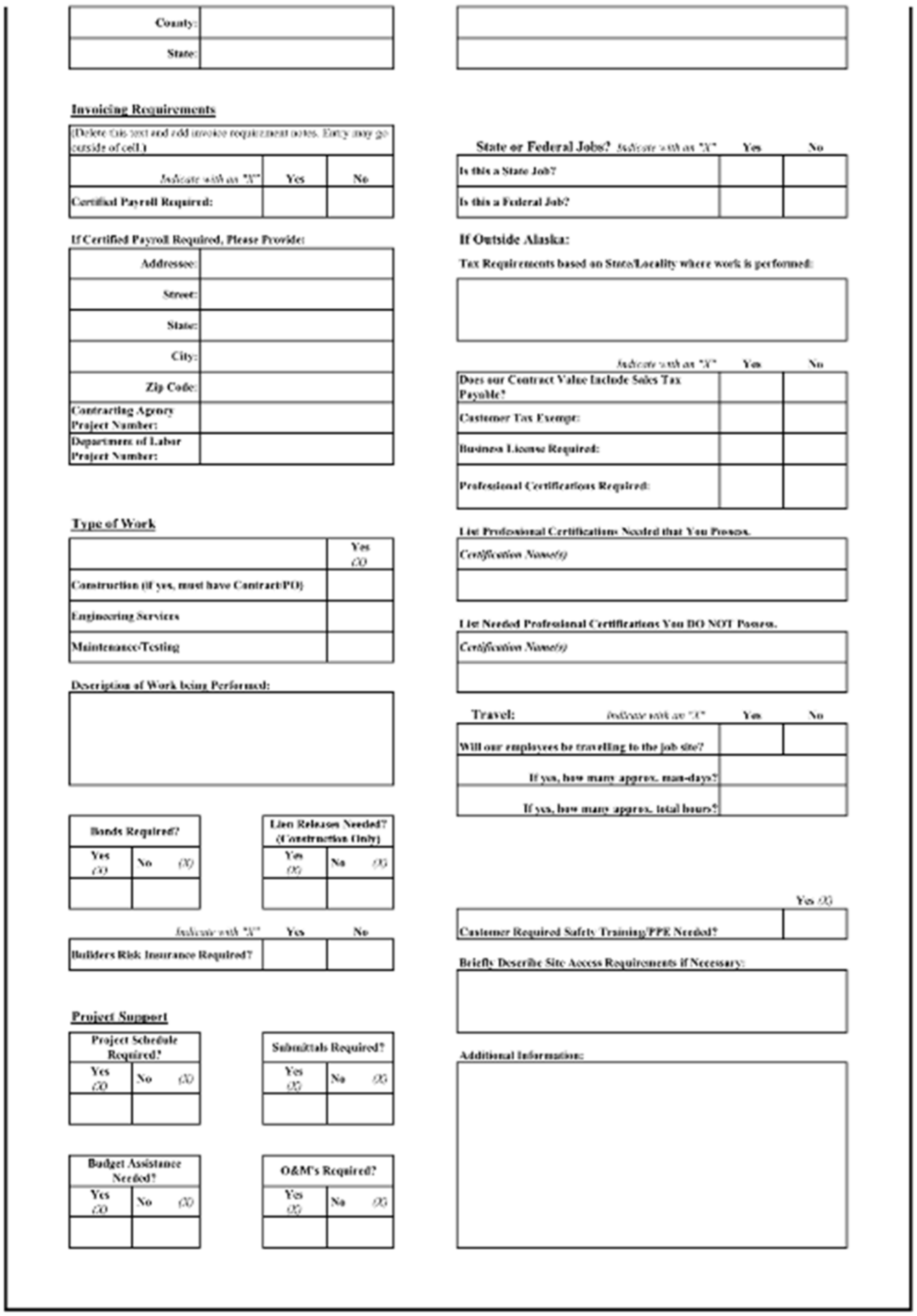
City:	
County:	
State:	

Project Location: (If different than Work Location)

City:	

Tasks: (i.e.: 01 Project Management)

(one per line, if larger list insert an additional sheet for task listing)



County:	
State:	

Invoicing Requirements

(Delete this text and add invoice requirement notes. Entry may go outside of cell.)		
Indicate with an "X"	Yes	No
Certified Payroll Required:		

If Certified Payroll Required, Please Provide:

Addressee:	
Street:	
State:	
City:	
Zip Code:	
Contracting Agency Project Number:	
Department of Labor Project Number:	

Type of Work

	Yes (X)
Construction (if yes, must have Contract/PO)	
Engineering Services	
Maintenance/Testing	

Description of Work being Performed:

Bonds Required?	
Yes (X)	No (X)

Lien Releases Needed? (Construction Only)	
Yes (X)	No (X)

Indicate with "X"	Yes	No
Builders Risk Insurance Required?		

Project Support

Project Schedule Required?	
Yes (X)	No (X)

Submittals Required?	
Yes (X)	No (X)

Budget Assistance Needed?	
Yes (X)	No (X)

O&M's Required?	
Yes (X)	No (X)

State or Federal Jobs? *Indicate with an "X"*	Yes	No
Is this a State Job?		
Is this a Federal Job?		

If Outside Alaska:

Tax Requirements based on State/Locality where work is performed:

Indicate with an "X"	Yes	No
Does our Contract Value Include Sales Tax Payable?		
Customer Tax Exempt:		
Business License Required:		
Professional Certifications Required:		

List Professional Certifications Needed that You Possess.

Certification Name(s)

List Needed Professional Certifications You DO NOT Possess.

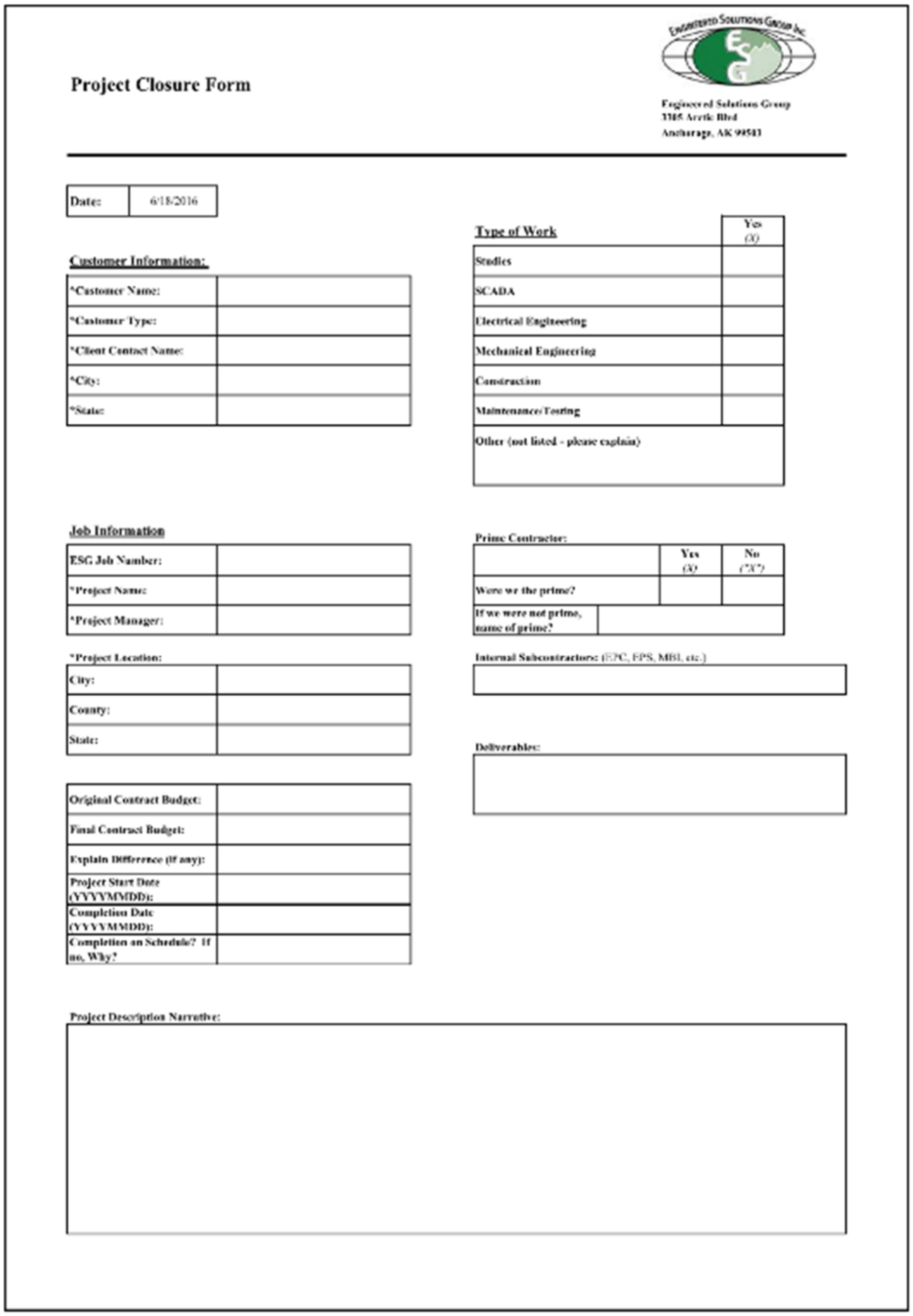
Certification Name(s)

Travel: *Indicate with an "X"*	Yes	No
Will our employees be travelling to the job site?		
If yes, how many approx. man-days?		
If yes, how many approx. total hours?		

	Yes (X)
Customer Required Safety Training/PPE Needed?	

Briefly Describe Site Access Requirements if Necessary:

Additional Information:



Project Closure Form

Engineered Solutions Group
3305 Arctic Blvd
Anchorage, AK 99503

Date:	6/18/2016

Type of Work

	Yes (X)
Studies	
SCADA	
Electrical Engineering	
Mechanical Engineering	
Construction	
Maintenance/Touring	
Other (not listed - please explain)	

Customer Information:

*Customer Name:	
*Customer Type:	
*Client Contact Name:	
*City:	
*State:	

Job Information

ESG Job Number:	
*Project Name:	
*Project Manager:	

*Project Location:

City:	
County:	
State:	

Original Contract Budget:	
Final Contract Budget:	
Explain Difference (if any):	
Project Start Date (YYYYMMDD):	
Completion Date (YYYYMMDD):	
Completion on Schedule? If no, Why?	

Prime Contractor:

	Yes (X)	No ("X")
Were we the prime?		
If we were not prime, name of prime?		

Internal Subcontractors: (E2C, EPS, MBI, etc.)

Deliverables:

Project Description Narrative:

Appendix B

Data Dictionary

Table B1. Definitions for *Project* Database Table by Field Name

Name	**Type**	**Description**
*id	int(10)	Unique ID for the record (auto increment)
*job_no	char(7)	ESG job number (##-####)
job_name	varchar(255)	Full project name (no acronyms)
client_name	varchar(50)	Client name
esg_comp	char(3)	ESG contract holding company (D&L, EPC, EPS, MBI, PBI, SEI)
prime_name	varchar(50)	Name of prime contractor (ESG for EPC, EPS, MBI, etc.)
int_subs	varchar(50)	Internal subcontractors (EPC, EPS, MBI, etc.)
pm_first	varchar(50)	ESG project manager first name
pm_last	varchar(50)	ESG project manager last name
contact_first	varchar(50)	Client contact first name
contact_last	varchar(50)	Client contact last name
state_job	varchar(1)	Is a state job (Y for yes, N for no, null for unknown)
fed_job	varchar(1)	Is a federal job (Y for yes, N for no, null for unknown)
contract_type	varchar(25)	Contract type (Time and Materials, Firm Fixed Price, or Unit Price)
org_budget	decimal(12,2)	Original contract budget
fin_budget	decimal(12,2)	Final contract budget
budget_exp	varchar(500)	Explanation of difference in budget
schedule_exp	varchar(500)	Project completed on schedule (Yes or No) and explanation of why not
deliverables	varchar(8000)	Project deliverables provided to the client (drawings, specifications, etc.)

proj_desc_clos	varchar(8000)	Project description narrative from project closure form
work_desc_crea	varchar(8000)	Description of work being performed from job creation form
add_info_crea	varchar(8000)	Additional information from job creation form (not always applicable)
notes	varchar(8000)	Project notes (not displayed in user interface)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B2. Definitions for *Contact* Database Table by Field Name

Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*first_name	varchar(50)	Contact first name
*last_name	varchar(50)	Contact last name
*comp_name	varchar(50)	Company name (matches client_name in project table)
contact_phone1	varchar(25)	Contact phone number 1
contact_phone2	varchar(25)	Contact phone number 2
contact_email	varchar(50)	Contact email address
notes	varchar(8000)	Contact notes (not displayed in user interface)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B3. Definitions for *Dates* Database Table by Field Name

Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*job_no	char(7)	ESG job number (##-####)
*event	varchar(50)	Event name
*event_date	date	Date of event (YYYY-MM-DD)
notes	varchar(8000)	Date notes (not displayed in user interface)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B4. Definitions for *Location* Database Table by Field Name

Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*name	varchar(50)	Location/facility name (where the work is being done for)
address	varchar(255)	Address (where the work is being done for)
city	varchar(50)	City (where the work is being done for)
county	varchar(50)	County (where the work is being done for)
state	varchar(50)	State (where the work is being done for)
zip	varchar(16)	Zip (where the work is being done for)
country	varchar(50)	Country (where the work is being done for)
description	varchar(8000)	Description of location (where the work is being done for)
notes	varchar(8000)	Location notes (not displayed in user interface)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B5. Definitions for *Project Location* Database Table by Field Name

Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*job_no	char(7)	ESG job number (##-####)
*name	varchar(50)	Location/facility name (matches name in location table)
*no_pt	tinyint(4)	Number of point locations for job (not displayed but needed for code)
*no_line	tinyint(4)	Number of polyline locations for job (not displayed but needed for code)
*no_poly	tinyint(4)	Number of polygon locations for job (not displayed but needed for code)
notes	varchar(8000)	Project location notes (not displayed in user interface)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B6. Definitions for *Component* Database Table by Field Name

Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*job_no	char(7)	ESG job number (##-####)
*component	varchar(50)	Name of component
manufacturer	varchar(50)	Manufacturer (if applicable)
component_desc	varchar(8000)	Description of component
lessons	varchar(8000)	Lessons learned regarding component (good and/or bad)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B7. Definitions for *Link* Database Table by Field Name

Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*job_no	char(7)	ESG job number (##-####)
*file_title	varchar(50)	Title for file
*link	varchar(8000)	File name with job number as prefix (ex: ##-####_xfmr.pdf)
file_desc	varchar(8000)	Description of file
component2	varchar(50)	Name of component (if applicable) (matches component in component table) (component2 for easier coding)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

Table B8. Definitions for *Type of Work* Database Table by Field Name

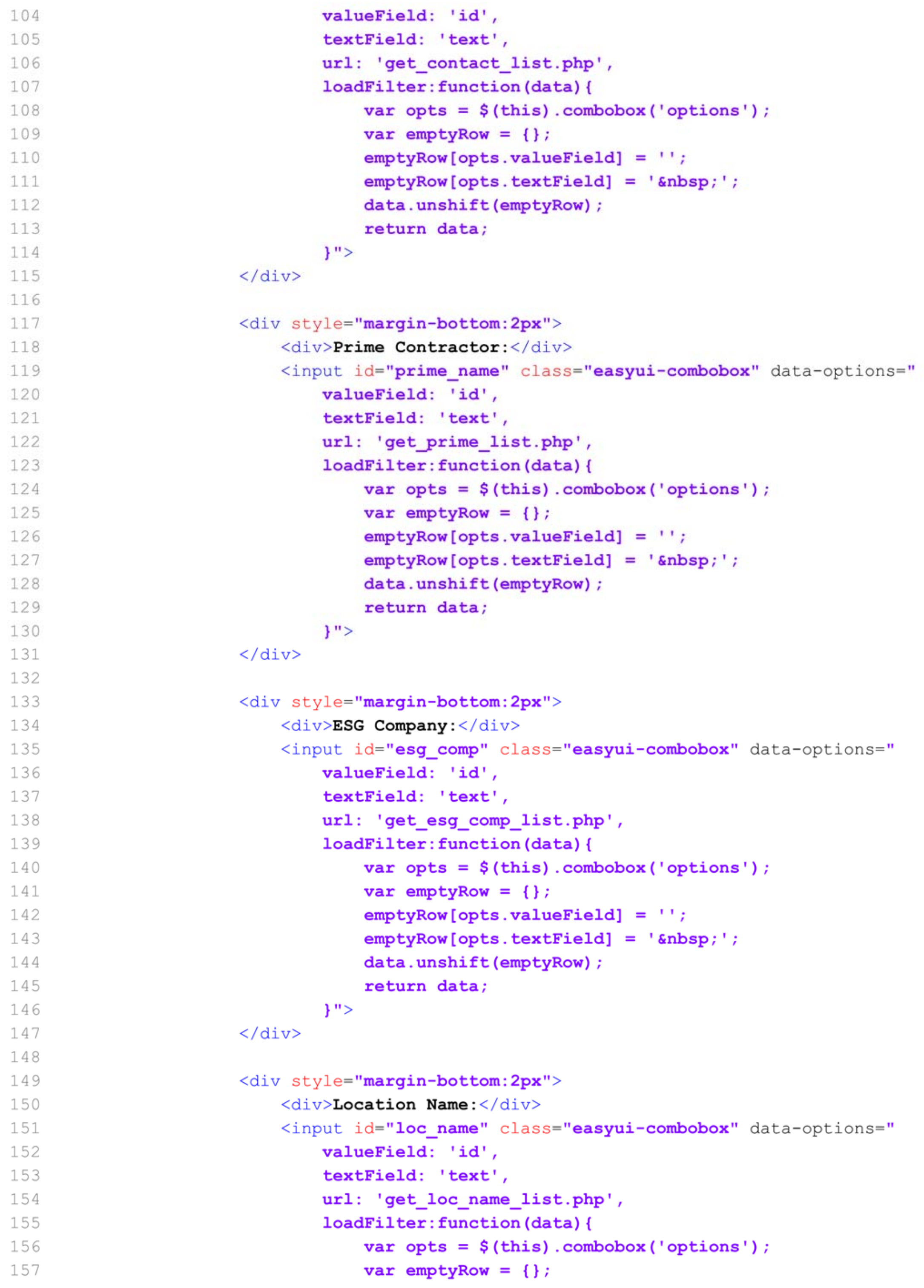
Name	Type	Description
*id	int(10)	Unique ID for the record (auto increment)
*job_no	char(7)	ESG job number (##-####)
*work_type	varchar(50)	Type of work (Studies, SCADA, Electrical Engineering, Mechanical Engineering, Construction, Maintenance/Testing, Other (specified))
notes	varchar(8000)	Type of work notes (not displayed in user interface)
created	timestamp	Time of creation (current time default)
modified	timestamp	Time of last modification (null default)

* Required field

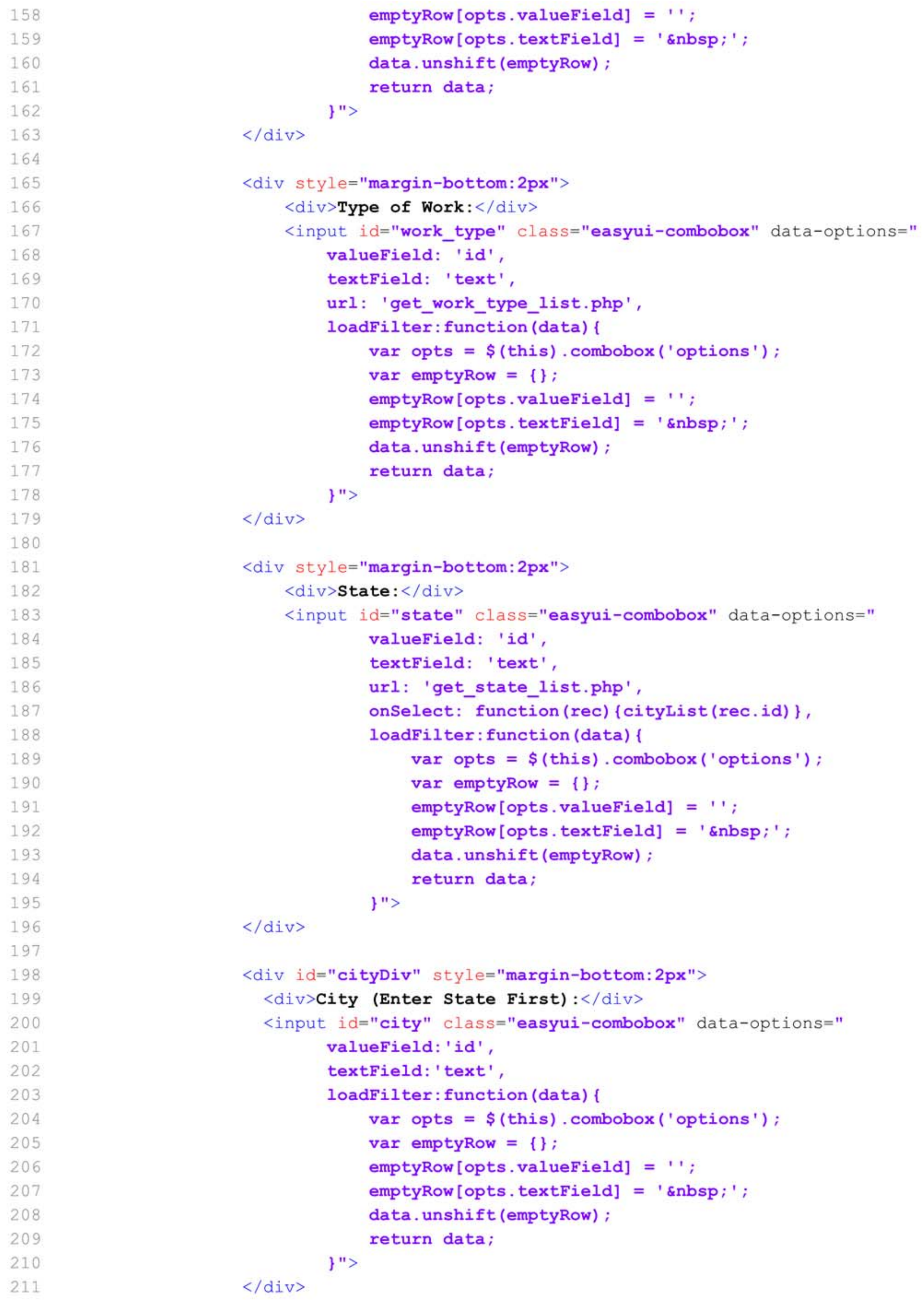
Appendix C

HTML Code

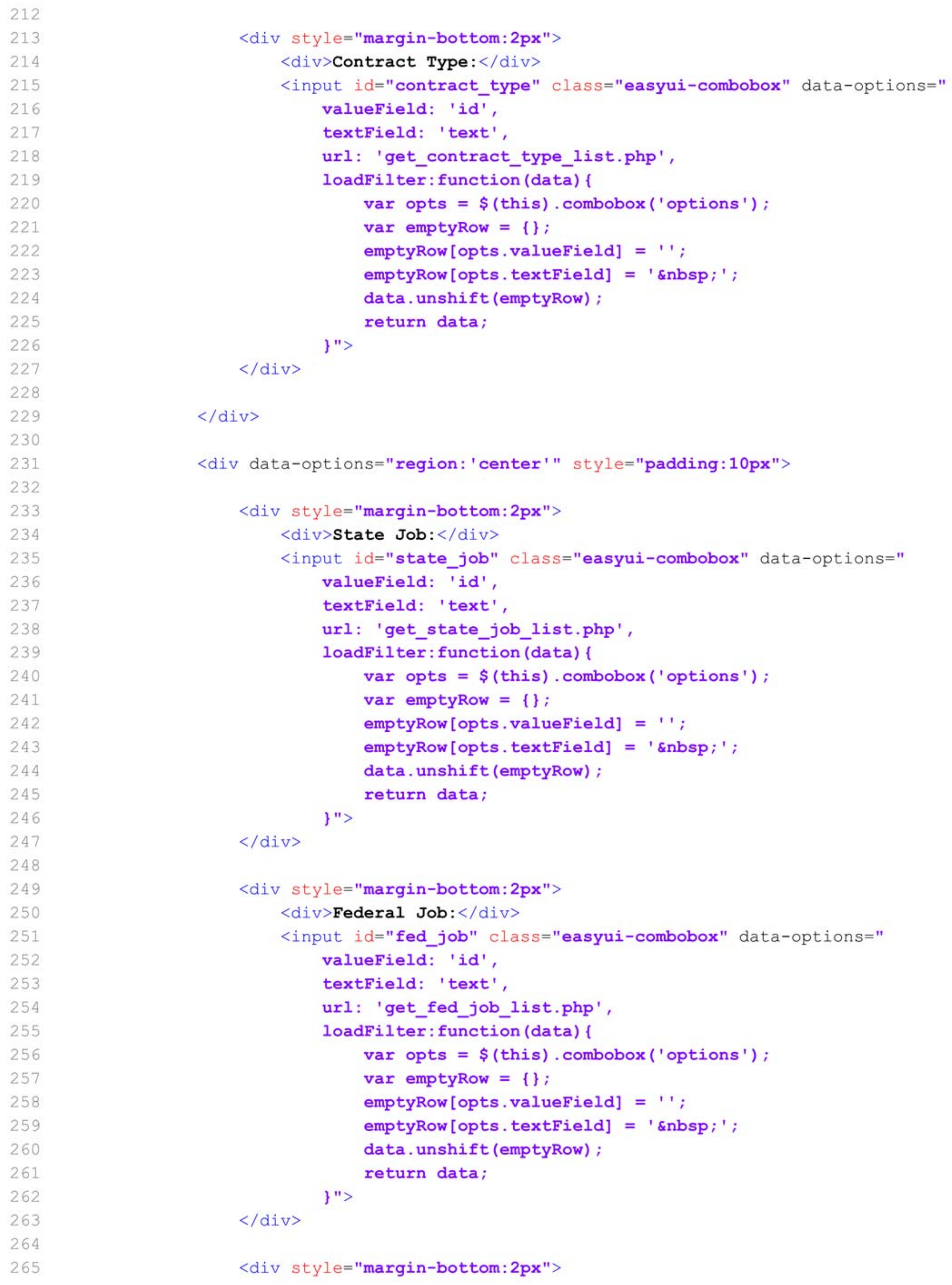
```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>ESG Projects</title>
<link rel="stylesheet" type="text/css" href="../jquery-easyui/themes/default/easyui.css">
    <style type="text/css">
  html { height: 100% }
      body { height: 100%; margin: 0px; padding: 0px }
      #map { height: 100% }
      input[type=button]{ margin: 5px }
      #sidebar { width: 90% }
      #msg { font-size: large }

  .labels {
     color: black;
     background-color: #FFFFE0;
     font-family: "Arial", sans-serif;
     font-size: 10px;
     text-align: center;
     border: 1px solid black;
     white-space: nowrap;
   }
    </style>

    <script
    src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
    <script>
        if (!window.jQuery) document.write('<script src="jquery-1.11.3.js"><\/script>');
    </script>
    <script type="text/javascript" src="../jquery-easyui/jquery.easyui.min.js"></script>
    <script src="https://maps.googleapis.com/maps/api/js"></script>
    <script src="markerwithlabel_packed.js"></script>

  </head>
  <body class="easyui-layout">
    <div id="map" data-options="region:'center',split:true"></div>
    <div id="dlg" class="easyui-dialog" title="Select Job Number or Job Criteria"
    closed="true" style="width:450px;padding:10px"
        data-options="iconCls:'icon-save',
        onClose:function() {
            $('#frmFilters').form('clear');
            cityList('');
        }" >

    <form id="frmFilters">
        <div id="dlgLayout" class="easyui-layout" data-options="fit:true"
        style="height:519px">
            <div data-options="region:'north'" style="padding:10px">

                <div style="margin-bottom:2px">
                    <div>Job Number:</div>
                    <input id="job_no" class="easyui-combobox" style="width:325px"
                    data-options="
                    valueField: 'id',
                    textField: 'text',
                    url: 'get_job_list.php',
                    loadFilter:function(data){
                        var opts = $(this).combobox('options');
                        var emptyRow = {};
                        emptyRow[opts.valueField] = '';
                        emptyRow[opts.textField] = '&nbsp;';
                        data.unshift(emptyRow);
                        return data;
                    }">

            <a href="javascript:void(0)" class="easyui-linkbutton" style="text-align:center" onclick="process()">Submit</a>
            </div>
        </div>

        <div data-options="region:'west'" style="width:210px;padding:10px">

            <div style="margin-bottom:2px">
                <div>Project Manager:</div>
                <input id="proj_mngr" class="easyui-combobox" data-options="
                    valueField: 'id',
                    textField: 'text',
                    url: 'get_pm_list.php',
                    loadFilter:function(data){
                        var opts = $(this).combobox('options');
                        var emptyRow = {};
                        emptyRow[opts.valueField] = '';
                        emptyRow[opts.textField] = '&nbsp;';
                        data.unshift(emptyRow);
                        return data;
                    }">
            </div>

            <div style="margin-bottom:2px">
                <div>Client:</div>
                <input id="client_name" class="easyui-combobox" data-options="
                    valueField: 'id',
                    textField: 'text',
                    url: 'get_client_list.php',
                    loadFilter:function(data){
                        var opts = $(this).combobox('options');
                        var emptyRow = {};
                        emptyRow[opts.valueField] = '';
                        emptyRow[opts.textField] = '&nbsp;';
                        data.unshift(emptyRow);
                        return data;
                    }">
            </div>

            <div style="margin-bottom:2px">
                <div>Client Contact:</div>
                <input id="contact" class="easyui-combobox" data-options="
```



```
                valueField: 'id',
                textField: 'text',
                url: 'get_contact_list.php',
                loadFilter:function(data){
                    var opts = $(this).combobox('options');
                    var emptyRow = {};
                    emptyRow[opts.valueField] = '';
                    emptyRow[opts.textField] = '&nbsp;';
                    data.unshift(emptyRow);
                    return data;
                }">
        </div>

        <div style="margin-bottom:2px">
            <div>Prime Contractor:</div>
            <input id="prime_name" class="easyui-combobox" data-options="
                valueField: 'id',
                textField: 'text',
                url: 'get_prime_list.php',
                loadFilter:function(data){
                    var opts = $(this).combobox('options');
                    var emptyRow = {};
                    emptyRow[opts.valueField] = '';
                    emptyRow[opts.textField] = '&nbsp;';
                    data.unshift(emptyRow);
                    return data;
                }">
        </div>

        <div style="margin-bottom:2px">
            <div>ESG Company:</div>
            <input id="esg_comp" class="easyui-combobox" data-options="
                valueField: 'id',
                textField: 'text',
                url: 'get_esg_comp_list.php',
                loadFilter:function(data){
                    var opts = $(this).combobox('options');
                    var emptyRow = {};
                    emptyRow[opts.valueField] = '';
                    emptyRow[opts.textField] = '&nbsp;';
                    data.unshift(emptyRow);
                    return data;
                }">
        </div>

        <div style="margin-bottom:2px">
            <div>Location Name:</div>
            <input id="loc_name" class="easyui-combobox" data-options="
                valueField: 'id',
                textField: 'text',
                url: 'get_loc_name_list.php',
                loadFilter:function(data){
                    var opts = $(this).combobox('options');
                    var emptyRow = {};
```



```
            emptyRow[opts.valueField] = '';
            emptyRow[opts.textField] = '&nbsp;';
            data.unshift(emptyRow);
            return data;
        }">
</div>

<div style="margin-bottom:2px">
    <div>Type of Work:</div>
    <input id="work_type" class="easyui-combobox" data-options="
        valueField: 'id',
        textField: 'text',
        url: 'get_work_type_list.php',
        loadFilter:function(data){
            var opts = $(this).combobox('options');
            var emptyRow = {};
            emptyRow[opts.valueField] = '';
            emptyRow[opts.textField] = '&nbsp;';
            data.unshift(emptyRow);
            return data;
        }">
</div>

<div style="margin-bottom:2px">
    <div>State:</div>
    <input id="state" class="easyui-combobox" data-options="
            valueField: 'id',
            textField: 'text',
            url: 'get_state_list.php',
            onSelect: function(rec){cityList(rec.id)},
            loadFilter:function(data){
                var opts = $(this).combobox('options');
                var emptyRow = {};
                emptyRow[opts.valueField] = '';
                emptyRow[opts.textField] = '&nbsp;';
                data.unshift(emptyRow);
                return data;
            }">
</div>

<div id="cityDiv" style="margin-bottom:2px">
  <div>City (Enter State First):</div>
  <input id="city" class="easyui-combobox" data-options="
        valueField:'id',
        textField:'text',
        loadFilter:function(data){
            var opts = $(this).combobox('options');
            var emptyRow = {};
            emptyRow[opts.valueField] = '';
            emptyRow[opts.textField] = '&nbsp;';
            data.unshift(emptyRow);
            return data;
        }">
</div>
```



```

                    <div style="margin-bottom:2px">
                        <div>Contract Type:</div>
                        <input id="contract_type" class="easyui-combobox" data-options="
                            valueField: 'id',
                            textField: 'text',
                            url: 'get_contract_type_list.php',
                            loadFilter:function(data){
                                var opts = $(this).combobox('options');
                                var emptyRow = {};
                                emptyRow[opts.valueField] = '';
                                emptyRow[opts.textField] = '&nbsp;';
                                data.unshift(emptyRow);
                                return data;
                            }">
                    </div>

                </div>

                <div data-options="region:'center'" style="padding:10px">

                    <div style="margin-bottom:2px">
                        <div>State Job:</div>
                        <input id="state_job" class="easyui-combobox" data-options="
                            valueField: 'id',
                            textField: 'text',
                            url: 'get_state_job_list.php',
                            loadFilter:function(data){
                                var opts = $(this).combobox('options');
                                var emptyRow = {};
                                emptyRow[opts.valueField] = '';
                                emptyRow[opts.textField] = '&nbsp;';
                                data.unshift(emptyRow);
                                return data;
                            }">
                    </div>

                    <div style="margin-bottom:2px">
                        <div>Federal Job:</div>
                        <input id="fed_job" class="easyui-combobox" data-options="
                            valueField: 'id',
                            textField: 'text',
                            url: 'get_fed_job_list.php',
                            loadFilter:function(data){
                                var opts = $(this).combobox('options');
                                var emptyRow = {};
                                emptyRow[opts.valueField] = '';
                                emptyRow[opts.textField] = '&nbsp;';
                                data.unshift(emptyRow);
                                return data;
                            }">
                    </div>

                    <div style="margin-bottom:2px">
```

```
                        <div>Search Project &amp; Work Desc.:</div>
                        <input class="easyui-textbox" id = "proj_desc" name="search">
                    </div>

                    <div style="margin-bottom:2px">
                        <div>Project Component:</div>
                        <input id="component" class="easyui-combobox" data-options="
                            valueField: 'id',
                            textField: 'text',
                            url: 'get_component_list.php',
                            loadFilter:function(data){
                                var opts = $(this).combobox('options');
                                var emptyRow = {};
                                emptyRow[opts.valueField] = '';
                                emptyRow[opts.textField] = '&nbsp;';
                                data.unshift(emptyRow);
                                return data;
                            }">
                    </div>

                    <div style="margin-bottom:2px">
                        <div>Search Component Desc.:</div>
                        <input class="easyui-textbox" id = "component_desc" name="search2">
                    </div>

                    <div style="margin-bottom:2px">
                        <div>Search Component Lessons Learned:</div>
                        <input class="easyui-textbox" id = "lessons" name="search3">
                    </div>

                    <div style="margin-bottom:2px">
                        <div>Manufacturer:</div>
                        <input id="manufacturer" class="easyui-combobox" data-options="
                            valueField: 'id',
                            textField: 'text',
                            url: 'get_manufacturer_list.php',
                            loadFilter:function(data){
                                var opts = $(this).combobox('options');
                                var emptyRow = {};
                                emptyRow[opts.valueField] = '';
                                emptyRow[opts.textField] = '&nbsp;';
                                data.unshift(emptyRow);
                                return data;
                            }">
                    </div>

                    <div style="margin-bottom:2px">
                        <div>File Title:</div>
                        <input id="file_title" class="easyui-combobox" data-options="
                            valueField: 'id',
                            textField: 'text',
                            url: 'get_file_title_list.php',
                            loadFilter:function(data){
                                var opts = $(this).combobox('options');
                        var emptyRow = {};
                        emptyRow[opts.valueField] = '';
                        emptyRow[opts.textField] = '&nbsp;';
                        data.unshift(emptyRow);
                        return data;
                    }">
            </div>

            <div style="margin-bottom:2px">
                <div>Search File Desc.:</div>
                <input class="easyui-textbox" id = "file_desc" name="search4">
            </div>

        </div>

        <div data-options="region:'south'"
        style="height:50px;text-align:center;padding:5px">
            <a href="javascript:void(0)" class="easyui-linkbutton"
            onclick="process()">Submit</a>
        </div>
    </div>
</form>

</div>
<div data-options="region:'south',split:true" title="Jobs" style="height:230px;">

  <fieldset>
    <input type="button" id="btnShow" value="Show All Jobs"
    onclick="clearFilterList();render('get_location.php?job_no=all')" />
    <input type="button" id="btnFilter" value="Select Job Criteria"
    onclick="$('#dlg').dialog('open')" />
    <div id="filters"></div>
  </fieldset>

  <div id="messageArea">
  </div>

  <table id="sidebar" class="easyui-datagrid" rownumbers="true" sortName="job_no"
  sortOrder="asc" remoteSort="false"
    striped="true" singleSelect="true">
    <thead>
      <tr>
        <th field="job_no" sortable="true">Job No.</th>
        <th field="layer" hidden="true">Layer</th>
        <th field="name" sortable="true">Job Name</th>
        <th field="location" sortable="true">Location</th>
        <th field="client_name" sortable="true">Client</th>
        <th field="pmngr" sortable="true">Project Manager</th>
        <th field="state" sortable="true">State</th>
        <th field="num" hidden="true">Num</th>
      </tr>
    </thead>
    <tbody>
    </tbody>
  </table>
</div>
<div id="dlg2" class="easyui-dialog" title="Project Data" closed="true" style="width:600px;height:500px;padding:10px">
</div>
</div>
<div id="dlg3" class="easyui-dialog" title="Job Search" data-options="modal:true" closed="true" style="width:200px;height:75px;padding:10px">
    No Matching Records.
</div>
<script src="dbf.js"></script>
<script src="shp.js"></script>
<script type="text/javascript">
  var infowindow;
  var map;
  var job_no="";
  var arrShp = [];
  var arrDbf = [];
  var shpfiles = [];
  var mapBounds;
  var markers = [];
  var polylines = [];
  var polygons = [];
  var shapeTypes;

  //Validates combobox values entered by user
  $.extend($.fn.validatebox.defaults.rules,{
           inList:{
                  validator:function(value,param){
                         var c = $(param[0]);
                         var opts = c.combobox('options');
                         var data = c.combobox('getData');
                         var exists = false;
                         for(var i=0; i<data.length; i++){
                                if (value == data[i][opts.textField]){
                                       exists = true;
                                       break;
                                }
                         }
                         return exists;
                  },
                  message:'Invalid Value'
           }
    })
    $('#job_no').combobox({
      validType:'inList["#job_no"]'
    });
    $('#proj_mngr').combobox({
      validType:'inList["#proj_mngr"]'
    });
    $('#client_name').combobox({
      validType:'inList["#client_name"]'
    });
    $('#contact').combobox({
            validType:'inList["#contact"]'
          });
          $('#prime_name').combobox({
            validType:'inList["#prime_name"]'
          });
          $('#esg_comp').combobox({
            validType:'inList["#esg_comp"]'
          });
          $('#loc_name').combobox({
            validType:'inList["#loc_name"]'
          });
          $('#work_type').combobox({
            validType:'inList["#work_type"]'
          });
          $('#state').combobox({
            validType:'inList["#state"]'
          });
          $('#city').combobox({
            validType:'inList["#city"]'
          });
          $('#contract_type').combobox({
            validType:'inList["#contract_type"]'
          });
          $('#state_job').combobox({
            validType:'inList["#state_job"]'
          });
          $('#fed_job').combobox({
            validType:'inList["#fed_job"]'
          });
          $('#component').combobox({
            validType:'inList["#component"]'
          });
          $('#manufacturer').combobox({
            validType:'inList["#manufacturer"]'
          });
          $('#file_title').combobox({
            validType:'inList["#file_title"]'
          });

        //Creates the map, loads in the SHP and DBF files
        function initMap() {
          markers = [];
          polylines = [];
          polygons = [];
          shapeTypes = new Array();
          shapeTypes[1] = "Point";
          shapeTypes[3] = "Polyline";
          shapeTypes[5] = "Polygon";
          map = new google.maps.Map(document.getElementById('map'), {
            zoom: 4,
            center: new google.maps.LatLng(63, -165),
            mapTypeId: google.maps.MapTypeId.TERRAIN
          });

        shpfiles = new Array('Point','Polyline','Polygon');

        for (var k=0; k < shpfiles.length; k++) {
            shpfile = shpfiles[k];
            SHPParser.load(shpfile + '.shp', shpLoad, shpLoadError);
            DBFParser.load(shpfile + '.dbf', dbfLoad, dbfLoadError);
        }

        if (!google.maps.Polygon.prototype.getBounds) {
          google.maps.Polygon.prototype.getBounds = function() {
              var bounds = new google.maps.LatLngBounds();
              var paths = this.getPaths();
              var path;
              for (var p = 0; p < paths.getLength(); p++) {
                  path = paths.getAt(p);
                  for (var i = 0; i < path.getLength(); i++) {
                      bounds.extend(path.getAt(i));
                  }
              }
              return bounds;
          }
        }

        if (!google.maps.Polyline.prototype.getBounds) {
          google.maps.Polyline.prototype.getBounds = function() {
              var bounds = new google.maps.LatLngBounds();
              var path = this.getPath();
              for (var i = 0; i < path.getLength(); i++) {
                  bounds.extend(path.getAt(i));
              }
              return bounds;
          }
        }
      }

       //Handles the callback from loading SHPParser by assigning the shp to a global
      function shpLoad(sh) {
        arrShp.push(sh);
        if (arrDbf.length == shpfiles.length && arrShp.length == shpfiles.length) {
            reorder();
        }
      }

      //Error handler for shploader
      function shpLoadError() {
        window.console.log('shp file failed to load');
      }

      //Handles the callback from loading DBFParser by assigning the dbf to a global
      function dbfLoad(db) {
        arrDbf.push(db);
        if (arrDbf.length == shpfiles.length && arrShp.length == shpfiles.length) {
            reorder();
        }
        }

        //Error handler for dbfloader
        function dbfLoadError() {
          console.log('dbf file failed to load');
        }

        //Files could be read in out of original sequence; this puts them back in sequence
        function reorder() {
          var fixedShp = [];
          var fixedDbf = [];

          for (var l = 0; l < shpfiles.length; l++) {
            name = shpfiles[l];

            for (var m = 0; m < arrShp.length; m++) {
              shp = arrShp[m];
              if (shp.fileName == (name + '.shp')) {
                  fixedShp[l] = shp;
                  break;
              }
            }

            for (var m = 0; m < arrDbf.length; m++) {
              dbf = arrDbf[m];
              if (dbf.fileName == (name + '.dbf')) {
                  fixedDbf[l] = dbf;
                  break;
              }
            }
          }

          arrShp = fixedShp;
          arrDbf = fixedDbf;
        }

        //Populates city list
        function cityList(state){
          var url = 'get_city_list.php?state='+state;
          $('#city').combobox('reload', url);
        }

        //Gets user entered values and passes them to render function, creates filter list
        function process() {
          //Gets user inputed combobox values
          job_no = $('#job_no').combobox('getValue');
          var client_name = encodeURIComponent($('#client_name').combobox('getValue'));
          var prime_name = encodeURIComponent($('#prime_name').combobox('getValue'));
          var esg_comp = encodeURIComponent($('#esg_comp').combobox('getValue'));
          var name = encodeURIComponent($('#loc_name').combobox('getValue'));
          var work_type = encodeURIComponent($('#work_type').combobox('getValue'));
          var state = encodeURIComponent($('#state').combobox('getValue'));
          var city = encodeURIComponent($('#city').combobox('getValue'));
          var contract_type = encodeURIComponent($('#contract_type').combobox('getValue'));
```

```
var state_job = encodeURIComponent($('#state_job').combobox('getValue'));
var fed_job = encodeURIComponent($('#fed_job').combobox('getValue'));
var component = encodeURIComponent($('#component').combobox('getValue'));
var manufacturer = encodeURIComponent($('#manufacturer').combobox('getValue'));
var file_title = encodeURIComponent($('#file_title').combobox('getValue'));
var pm_cont = $('#proj_mngr').combobox('getValue');

//Splits project manager into two variables using comma
if (pm_cont !="") {
    var res = pm_cont.split(", ");
    var last = res[0];
    var first = res[1];
} else {
    var last = "";
    var first = "";
}
//Splits client contact into two variables using comma
var cnt_cont = $('#contact').combobox('getValue');
if (cnt_cont !="") {
    var res = cnt_cont.split(", ");
    var cnt_last = res[0];
    var cnt_first = res[1];
} else {
    var cnt_last = "";
    var cnt_first = "";
}

//Gets user inputed textbox values
var keyword = $('#proj_desc').textbox('getValue');
var keyword2 = $('#component_desc').textbox('getValue');
var keyword3 = $('#lessons').textbox('getValue');
var keyword4 = $('#file_desc').textbox('getValue');
var url = "get_location.php";

//Creates a filter list based on user entered values, capitalizes field names and replaces underscores with spaces
var fieldsCombo = ["job_no", "proj_mngr", "client_name", "contact", "prime_name", "esg_comp", "loc_name", "work_type",
"state", "city", "contract_type", "state_job", "fed_job", "component", "manufacturer", "file_title"];
var url2 = "get_location.php?";
var filterList = "";

for (i = 0; i < fieldsCombo.length; i++) {
  var field = fieldsCombo[i];
  var val = $('#' + field).combobox('getValue');
  url2 += field + "=" + val + "&";
  if (val) {
    field = field.replace("_", " ");
    field = titleCase(field);
    filterList += field + ": " + val + " | ";
  }
}
var fieldsText = ["proj_desc", "component_desc", "lessons", "file_desc"];
```

```
     for (i = 0; i < fieldsText.length; i++) {
      var field = fieldsText[i];
      var val = $('#' + field).textbox('getValue');
      url2 += field + "=" + val + "&";
      if (val) {
       field = field.replace("_", " ");
       field = titleCase(field);
       filterList += field + ": " + val + " | ";
      }
     }
     filterList = filterList.slice(0, -2); //Trim off extra pipe
     url2 = url2.slice(0, -1); //Trim off extra ampersand
     if (filterList != "") {
       $('#filters').html("Currently filtering by...<br>" + filterList);
     } else {
       $('#filters').html("");
     }

     //URL to pass to render function
     url += "?job_no=" + job_no + "&esg_comp=" + esg_comp + "&name=" + name
     + "&work_type=" + work_type + "&state=" + state + "&city=" + city +
     "&client_name=" + client_name
     + "&contract_type=" + contract_type + "&state_job=" + state_job + "&fed_job=" +
     fed_job
     + "&proj_desc_clos=" + keyword + "&work_desc_crea=" + keyword + "&pm_last=" +
     last + "&pm_first="
     + first + "&contact_last=" + cnt_last + "&contact_first=" + cnt_first +
     "&prime_name=" + prime_name
     + "&component=" + component + "&manufacturer=" + manufacturer + "&file_title="
     + file_title
     + "&component_desc=" + keyword2 + "&lessons=" + keyword3 + "&file_desc=" +
     keyword4;

     render(url);
     //Clears combobox and textbox values
     clearLists();
     //Closes Select Job Criteria dialog
     $('#dlg').dialog('close');
    }

    //Capitalizes field names for filter list
    function titleCase(str) {
     var newstr = new Array();
     if (str.indexOf(" ")>0){
       newstr = str.split(" ");
     } else {
       newstr[0]=str;
     }
     for(j=0;j<newstr.length;j++){
       var copy = newstr[j].substring(1).toLowerCase();
       newstr[j] = newstr[j][0].toUpperCase() + copy;
     }
     newstr = newstr.join(" ");
     return newstr;
```

```

}

//Clears combobox and textbox values
function clearLists() {
  $('#job_no').combobox('clear');
  $('#proj_mngr').combobox('clear');
  $('#client_name').combobox('clear');
  $('#contact').combobox('clear');
  $('#prime_name').combobox('clear');
  $('#esg_comp').combobox('clear');
  $('#loc_name').combobox('clear');
  $('#work_type').combobox('clear');
  $('#state').combobox('clear');
  $('#city').combobox('clear');
  $('#contract_type').combobox('clear');
  $('#state_job').combobox('clear');
  $('#fed_job').combobox('clear');
  $('#proj_desc').textbox('clear');
  $('#component').combobox('clear');
  $('#component_desc').textbox('clear');
  $('#lessons').textbox('clear');
  $('#manufacturer').combobox('clear');
  $('#file_title').combobox('clear');
  $('#file_desc').textbox('clear');
}

//Clears "Currently filtering by..." text between buttons and table
function clearFilterList() {
  $('#filters').html("");
}

//Adds overlays for all features in the shapefile
function render(url) {
  showLoading();
  clearMap();
  mapBounds = new google.maps.LatLngBounds();
  var pts2Add; var lines2Add; var polys2Add; var ptInfo; var lineInfo; var polyInfo;
  var ptJobNums; var lineJobNums; var polyJobNums; var ptLocName; var lineLocName; var polyLocName;
  var accordionDefBegin = '<input type="button" value="Project Data" onclick="openDialog(\'get_job_info.php?job_number=';
  var accordionDefEnd = '\')" />';

  $.ajax({ url: url, dataType: "xml", success: function(xmlDoc){
    if ($(xmlDoc).find("location").length==0){$('#dlg3').dialog('open');}
    hideLoading();
    $(xmlDoc).find("location").each(function(){
      var location = $(this);
      var name = $(location).attr("name");

      pts2Add = new Array();
      lines2Add = new Array();
```

```
            polys2Add = new Array();
            ptInfo = new Array();
            lineInfo = new Array();
            polyInfo = new Array();
            ptJobNums = new Array();
            lineJobNums = new Array();
            polyJobNums = new Array();
            ptLocName = new Array();
            lineLocName = new Array();
            polyLocName = new Array();

            $(location).find("job").each(function() {
                var arrGeomsFlag = new Array();
                arrGeomsFlag[0] = $(this).attr("pt");
                arrGeomsFlag[1] = $(this).attr("line");
                arrGeomsFlag[2] = $(this).attr("poly");

                for (var x = 0; x <= 2; x++) {
                    if (arrGeomsFlag[x] > 0) {
                      shp = arrShp[x]; dbf = arrDbf[x];

                      for (var p = 0; p < shp.records.length; p++) {
                        var dbfRec = dbf.records[p];
                        if (dbfRec["Name"]==name) {
                            var job = $(this).attr("no");
                            var client_name = $(this).attr("client_name");
                            var pm_last = $(this).attr("pm_last");
                            var pm_first = $(this).attr("pm_first");
                            var pmngr = pm_last.concat(", ", pm_first);
                            var state = $(this).attr("state");
                            var job_name = $(this).attr("job_name");

                            if (job_no == "" || job == job_no) {

                                addToSidebar(shp.records[p].shape.type,job,job_name,name,
                                client_name,pmngr,state,p);
                            }

                            var info = "Job#: " + job + "<br>Client: " + client_name +
                            "<br>PM: " + pmngr + "<br>State: " + state +
                                "<br>Job name: " + job_name;

                            if (x == 0) {
                              pts2Add[p] = shp.records[p].shape;
                              if (ptInfo[p]) { ptInfo[p] += "<hr>" + info; } else {
                              ptInfo[p] = info; }
                              ptInfo[p] += "<br>" + accordionDefBegin + job +
                              accordionDefEnd;
                              ptJobNums[p] = job;
                              ptLocName[p] = name;
                            } else if (x == 1) {
                              lines2Add[p] = shp.records[p].shape;
                              if (lineInfo[p]) { lineInfo[p] += "<hr>" + info; } else {
                              lineInfo[p] = info; }
                        lineInfo[p] += "<br>" + accordionDefBegin + job + accordionDefEnd;
                        lineJobNums[p] = job;
                        lineLocName[p] = name;
                      } else if (x == 2) {
                        polys2Add[p] = shp.records[p].shape;
                        if (polyInfo[p]) { polyInfo[p] += "<hr>" + info; } else { polyInfo[p] = info; }
                        polyInfo[p] += "<br>" + accordionDefBegin + job + accordionDefEnd;
                        polyJobNums[p] = job;
                        polyLocName[p] = name;
                      }
                  } // end if (dbf(Name) == name)
                } // end shapefile loop
              } // end if (arrGeoms[x] > 0)
            } // end 0-2 loop
      }); // end job element loop


      createMarkers(pts2Add,ptInfo,ptJobNums,ptLocName);
      createLines(lines2Add,lineInfo,lineJobNums,lineLocName);
      createPolys(polys2Add,polyInfo,polyJobNums,polyLocName);
    }); // end location element loop
  }}); // end reading of get_location output

} // end render function

//Clears info windows, shapefiles, and table
function clearMap() {
  if (typeof infowindow != 'undefined') {
      infowindow.close();
  }

  for (var key in markers) {
      markers[key].setMap(null);
  }

  for (var key in polylines) {
      polylines[key].setMap(null);
  }

  for (var key in polygons) {
      polygons[key].setMap(null);
  }

  $('#sidebar').datagrid('loadData', {"total":0,"rows":[]});
  $('#sidebar tr').remove();
}

//Shows Loading... message
function showLoading() {
  $("#messageArea").html("<h3>Loading...</h3>");
  $("#btnShow").prop("disabled", true);
```

```
      $("#btnFilter").prop("disabled", true);
    }

    //Hides Loading... message
    function hideLoading() {
      $("#messageArea").html("");
      $("#btnShow").prop("disabled", false);
      $("#btnFilter").prop("disabled", false);
    }

    //Opens Project Data dialog
    function openDialog(url) {
      $('#dlg2').dialog({
        href: url,
        modal: true
      });
      $('#dlg2').dialog('open');
    }

    //Adds table contents
    function addToSidebar(shapeType, job, job_name, name, client_name, pmngr, state,
    i) {
      var lastRow = $('<tr/>').appendTo($("#sidebar").find('tbody:last'));
      lastRow.append($('<td>').text(job));
      lastRow.append($('<td/>').text(shapeTypes[shapeType]));
      lastRow.append($('<td/>').text(job_name));
      lastRow.append($('<td/>').text(name));
      lastRow.append($('<td/>').text(client_name));
      lastRow.append($('<td/>').text(pmngr));
      lastRow.append($('<td/>').text(state));
      lastRow.append($('<td/>').text(i));
    }

    $('#sidebar').datagrid({
      onClickRow: function(index,row){
        myclick(row.layer, row.num);
      }
    });

    //Opens info window when table row is clicked
    function myclick(layer, num) {
      var arr;
      switch (layer) {
        case "Point": arr = markers; break;
        case "Polyline": arr = polylines; break;
        case "Polygon": arr = polygons; break;
      }
      google.maps.event.trigger(arr[num], "click",{});
    }

    //Creates points
    function createMarkers(pts2Add,ptInfo,ptJobNums,ptLocName) {
      for (var key in pts2Add) {
        var shape = pts2Add[key];
```

```
            var info = ptInfo[key];
            var job = ptJobNums[key];
            var name = ptLocName[key];
            var marker = new google.maps.Marker({
              position: new google.maps.LatLng(shape.content.y,shape.content.x),
              map: map,
              zIndex: (key * 100),
              title: name
            });

            markers[key] = marker;

            mapBounds.extend(new google.maps.LatLng(shape.content.y,shape.content.x));

            handle_clicks(marker, 1, info, job);
          }
        }

        //Creates lines with makePolyline function
        function createLines(lines2Add,lineInfo,lineJobNums,lineLocName) {
          for (var key in lines2Add) {
              var shape = lines2Add[key];
              var info = lineInfo[key];
              var job = lineJobNums[key];
              var name = lineLocName[key];
              points = pathToArray(shape.content.points);

              polyline = makePolyline(points, key, name);

              polylines[key] = polyline;

              mapBounds.union(polyline.getBounds());

              handle_clicks(polyline, 3, info, job);
          }
        }

        //Needed for creation of polylines and polygons
        function pathToArray(path) {
          var polygonPoints = [];
          for (var i = 0; i < path.length; i += 2) {
            polygonPoints.push(new google.maps.LatLng(path[i + 1], path[i]));
          }
          return polygonPoints;
        }

        //Creates lines with createLines function, adds marker for location name label
        function makePolyline(lineCoords, key, lineLabel) {
          var marker = new MarkerWithLabel({
              position: new google.maps.LatLng(0,0),
              draggable: false,
              raiseOnDrag: false,
              map: map,
              labelContent: lineLabel,
              labelAnchor: new google.maps.Point(30, 20),
              labelClass: "labels", // the CSS class for the label
              labelStyle: {opacity: 1.0},
              icon: "-text.png",
              visible: false
           });

          var line = new google.maps.Polyline({
              path: lineCoords,
              strokeColor: "#FF0000",
              strokeWeight: 2,
              map: map,
              zIndex: (key * 100)
          });

          google.maps.event.addListener(line, "mousemove", function(event) {
              marker.setPosition(event.latLng);
              marker.setVisible(true);
          });
          google.maps.event.addListener(line, "mouseout", function(event) {
              marker.setVisible(false);
          });
          return line;
        }

        //Creates polygons with makePolygon function
        function createPolys(polys2Add,polyInfo,polyJobNums,polyLocName) {
          for (var key in polys2Add) {
              var shape = polys2Add[key];
              var info = polyInfo[key];
              var job = polyJobNums[key];
              var name = polyLocName[key];
              var polygonPoints = [];
              var parts = shape.content.parts;
              if (parts.length === 1) {
                polygonPoints.push(pathToArray(shape.content.points));
              } else {
                var q;
                for (q = 0; q < parts.length - 1; q++) {
                  polygonPoints.push(pathToArray(shape.content.points.subarray(2 *
                  parts[q], 2 * parts[q + 1])));
                  if (2 * parts[q + 1] > shape.content.points.length) {
                    throw new Error('part index beyond points array end');
                  }
                }
              }

              polygon = makePolygon(polygonPoints, key, name);

              polygons[key] = polygon;

              mapBounds.union(polygon.getBounds());

              handle_clicks(polygon, 5, info, job);
```

```

            }
            google.maps.event.addListenerOnce(map, 'bounds_changed', function(event) {
                    if (this.getZoom() > 15){
                            this.setZoom(15);
                    }
            });

            map.fitBounds(mapBounds);
            hideLoading();
            $('#sidebar').datagrid();
          }

          //Creates polygons with createPolys function, adds marker for location name label
          function makePolygon(polyCoords, key, polyLabel) {
            var marker = new MarkerWithLabel({
                position: new google.maps.LatLng(0,0),
                draggable: false,
                raiseOnDrag: false,
                map: map,
                labelContent: polyLabel,
                labelAnchor: new google.maps.Point(30, 20),
                labelClass: "labels", // the CSS class for the label
                labelStyle: {opacity: 1.0},
                icon: "-text.png",
                visible: false
             });

            var poly = new google.maps.Polygon({
                paths: polyCoords,
                strokeColor: "#FF0000",
                strokeWeight: .3,
                strokeOpacity: 0.8,
                fillColor: "#FF0000",
                fillOpacity: .2,
                map: map,
                zIndex: (key * 100)
            });

            google.maps.event.addListener(poly, "mousemove", function(event) {
                marker.setPosition(event.latLng);
                marker.setVisible(true);
            });
            google.maps.event.addListener(poly, "mouseout", function(event) {
                marker.setVisible(false);
            });
            return poly;
          }

          //Displays an info window when a point, polyline, or polygon is clicked
          function handle_clicks(overlay, type, info, job) {
            google.maps.event.addListener(overlay, 'click', function(e) {
              if (typeof infowindow != 'undefined') {
                infowindow.close();
          }

          var pos;
          if (typeof e.latLng!="undefined") {
            pos = e.latLng;
          } else {
            switch (type) {
              case 1: pos = overlay.getPosition(); break;
              case 3: pos = getLineMidpoint(overlay); break;
              case 5: pos = getBoundsForPoly(overlay).getCenter(); break;
            }
          }

          infowindow = new google.maps.InfoWindow({
            content: info,
            position: pos,
            map: map
          });
        });
      }

      //Returns the LatLng midway along the array of LatLngs defining the Polyline
      function getLineMidpoint(line) {
        var path = line.getPath();
        var midpt = Math.round(path.getLength() / 2);
        return path.getAt(midpt);
      }

      //Determines location for info window for polygons
      function getBoundsForPoly(poly) {
        var bounds = new google.maps.LatLngBounds;
        poly.getPath().forEach(function(latLng) {
            bounds.extend(latLng);
        });
        return bounds;
      }

    google.maps.event.addDomListener(window,'load',initMap);
    </script>

  </body>
</html>
```

Appendix D

PHP Code – Get Location

```
<?php
//Gets locations and table data for all locations when Show All Jobs button is clicked and gets locations
//and table data for locations matching selected criteria when Select Job Criteria button is clicked
$job_no = $_REQUEST["job_no"];

$link = mysqli_connect("localhost","user","password","esg_projects");

//Arrays for each of the database fields with matching combobox and textbox values (if not hard coded below)
$cols1 = array("job_no","pm_first","pm_last","contact_first","contact_last","client_name","prime_name","esg_comp","contract_type","state_job","fed_job");
$cols2 = array("name","state","city");
$cols3 = array("work_type");
$cols4 = array("component","manufacturer");
$cols5 = array("file_title");
$cols6 = array("lessons");
$cols7 = array("file_desc");
//Gets all the unique location names matching user-inputed criteria
$selectclause = 'SELECT distinct project_location.name FROM project_location INNER JOIN project ON project_location.job_no=project.job_no
LEFT OUTER JOIN type_of_work ON project_location.job_no=type_of_work.job_no
LEFT OUTER JOIN component ON project_location.job_no=component.job_no
LEFT OUTER JOIN link ON project_location.job_no=link.job_no
INNER JOIN location ON project_location.name=location.name ';
$whereclause = '';
$orderbyclause = 'ORDER BY project_location.job_no';
$query = '';

if ($job_no <> "all") {
    foreach ($cols1 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "project.$col = '$val' AND ";
        }
    }

    foreach ($cols2 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "location.$col = '$val' AND ";
        }
    }

    foreach ($cols3 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "type_of_work.$col = '$val' AND ";
        }
    }
    $val = $_REQUEST["proj_desc_clos"];
    if (isset($val) AND $val!="") {
        $whereclause .= "(project.proj_desc_clos LIKE '%$val%' OR project.work_desc_crea LIKE '%$val%') AND ";
    }

    foreach ($cols4 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "component.$col = '$val' AND ";
        }
    }

    foreach ($cols5 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "link.$col = '$val' AND ";
        }
    }
    $val = $_REQUEST["component_desc"];
    if (isset($val) AND $val!="") {
        $whereclause .= "(component.component LIKE '%$val%' OR component.manufacturer LIKE '%$val%' OR component.component_desc LIKE '%$val%') AND ";
    }

    foreach ($cols6 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "component.$col LIKE '%$val%' AND ";
        }
    }

    foreach ($cols7 as $col) {
        $val = $_REQUEST[$col];
        if (isset($val) AND $val!="") {
            $whereclause .= "link.$col LIKE '%$val%' AND ";
        }
    }

    $whereclause = substr($whereclause, 0, -4);
}

$query = $selectclause;

if ($whereclause) { $query .= "WHERE $whereclause"; }

$query .= $orderbyclause;
$result = mysqli_query($link,$query);
if ($result !== 0) {
    header("Content-type: text/xml");
    echo '<jobs>';

    $num_results = mysqli_num_rows($result);
    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $project_loc = htmlspecialchars($row['name'],ENT_QUOTES);
```

```php
        echo '<location name="' .$project_loc. '">';
        //Gets additional data for unique location names matching criteria
        $sql = "SELECT project_location.job_no, project_location.name,
        project_location.no_pt,
        project_location.no_line, project_location.no_poly, project.client_name,
        project.pm_last,
        project.pm_first, location.state, project.job_name FROM project_location INNER
        JOIN project ON
        project_location.job_no=project.job_no INNER JOIN location ON
        project_location.name=location.name
        WHERE project_location.name = '" .$project_loc. "'";
        $rs = mysqli_query($link,$sql);
        $num_records = mysqli_num_rows($rs);
        for ($j=0;$j<$num_records;$j++) {
            $rec = mysqli_fetch_array($rs);
            $job_no = $rec['job_no'];
            $name = htmlspecialchars($rec['name'],ENT_QUOTES);
            $no_pt = $rec['no_pt'];
            $no_line = $rec['no_line'];
            $no_poly = $rec['no_poly'];
            $client_name = htmlspecialchars($rec['client_name'],ENT_QUOTES);
            $pm_last = $rec['pm_last'];
            $pm_first = $rec['pm_first'];
            $state = htmlspecialchars($rec['state'],ENT_QUOTES);
            $job_name = htmlspecialchars($rec['job_name'],ENT_QUOTES);

            //pt, line, & poly needed for render function
            echo '<job no="' .$job_no. '" name="' .$name. '" pt="' .$no_pt. '" line="'
            .$no_line. '" poly="' .$no_poly. '" client_name="' .$client_name. '"
            pm_last="' .$pm_last. '" pm_first="' .$pm_first. '" state="' .$state. '"
            job_name="' .$job_name. '" />';
        }
        echo '</location>';
    }

    echo '</jobs>';
} else {
    echo 'Problem with query!';
}

mysqli_close($link);
?>
```

Appendix E

PHP Code – Get Job Info

```php
<?php
//Gets Project Data for accordion dialog when Project Data button is clicked in
infowindow
$job_number = $_REQUEST["job_number"];

$link = mysqli_connect("localhost","user","password","esg_projects");

//Job#:##-#### accordion tab
$sql = "SELECT project.job_no, project.job_name, project.pm_first, project.pm_last,
project.state_job,
project.fed_job, project.contract_type, project.client_name
FROM project ";

if ($job_number <> "all") {
    $where = "WHERE project.job_no = '" .$job_number. "' ";
}

$orderBy = "ORDER BY project.job_no;";
$orderBy2 = "ORDER BY component.component;";
$orderBy3 = "ORDER BY link.file_title;";
$sql .= $where.$orderBy;

$result = mysqli_query($link,$sql);

if ($result !== 0) {

    $num_results = mysqli_num_rows($result);
    echo '<div class="easyui-accordion" style="width:500px;height:425px;">';

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $job_no = $row['job_no'];
        $job_name = htmlspecialchars($row['job_name'],ENT_QUOTES);
        $client_name = htmlspecialchars($row['client_name'],ENT_QUOTES);
        $pm_first = $row['pm_first'];
        $pm_last = $row['pm_last'];
        $state_job = $row['state_job'];
        $fed_job = $row['fed_job'];
        $contract_type = htmlspecialchars($row['contract_type'],ENT_QUOTES);
        echo '<div title="Job#: '.$job_no.'" style="overflow:auto;padding:10px;">';
            echo '<strong>Job Name: </strong> '.$job_name.' <p></p>';
            echo '<strong>Client Name: </strong> '.$client_name.' <p></p>';
            echo '<strong>Project Manager: </strong> '.$pm_first.' '.$pm_last.' <p></p>';
            echo '<strong>Contract Type: </strong> '.$contract_type.' <p></p>';

            if ($state_job != '') {
                echo '<strong>State Job?: </strong> '.$state_job.' <p></p>';
            }
            if ($fed_job != '') {
                echo '<strong>Federal Job?: </strong> '.$fed_job.' <p></p>';
            }
    }
    $sql = "SELECT type_of_work.work_type FROM type_of_work WHERE type_of_work.job_no=
'" .$job_number. "' ";
```

```

    $result = mysqli_query($link,$sql);

    if ($result !== 0) {

        $num_results = mysqli_num_rows($result);
        $work_type="";
        for ($i=0;$i<$num_results;$i++) {
            $row = mysqli_fetch_array($result);
            $work_type .= htmlspecialchars($row['work_type'],ENT_QUOTES) .", ";

        }
        $work_type=substr($work_type,0,-2);
        echo '<strong>Type of Work: </strong> '.$work_type.' <p></p>';
    }
    echo '</div>';
}

//Companies & Contacts accordion tab
$sql = "SELECT project.client_name, project.esg_comp, project.prime_name,
project.int_subs, project.pm_first,
project.pm_last, project.contact_first, project.contact_last, contact.contact_phone1,
contact.contact_phone2,
contact.contact_email FROM project
INNER JOIN contact ON project.contact_first=contact.first_name AND
project.contact_last=contact.last_name ";

$sql .= $where.$orderBy;

$result = mysqli_query($link,$sql);

if ($result !== 0) {

    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $prime_name = htmlspecialchars($row['prime_name'],ENT_QUOTES);
        $esg_comp = $row['esg_comp'];
        $int_subs = $row['int_subs'];
        $contact_first = $row['contact_first'];
        $contact_last = $row['contact_last'];
        $contact_phone1 = $row['contact_phone1'];
        $contact_phone2 = $row['contact_phone2'];
        $contact_email = htmlspecialchars($row['contact_email'],ENT_QUOTES);

        echo '<div title="Companies & Contacts" style="padding:10px;">';
            if ($prime_name != '') {
                echo '<strong>Prime Contractor: </strong> '.$prime_name.' <p></p>';
            }
            echo '<strong>ESG Company: </strong> '.$esg_comp.' <p></p>';
            echo '<strong>Project Manager: </strong> '.$pm_first.' '.$pm_last.' <p></p>';
            if ($int_subs != '') {
                echo '<strong>Internal Subs: </strong> '.$int_subs.' <p></p>';
            }
            echo '<strong>Client Name: </strong> '.$client_name.' <p></p>';
            if ($contact_last != '') {
            echo '<strong>Client Contact: </strong> '.$contact_first.'
            '.$contact_last.' <p></p>';
            }
            if ($contact_email != '') {
                echo '<strong>Contact Email: </strong> '.$contact_email.' <p></p>';
            }
            if ($contact_phone1 != '') {
                echo '<strong>Contact Phone: </strong> '.$contact_phone1.' <p></p>
                '.$contact_phone2.'';
            }
        echo '</div>';
    }
}

//Location:XXX accordion tab(s)
$sql = "SELECT project_location.name, location.address, location.city, location.county,
location.state,
location.zip, location.country, location.description FROM project
INNER JOIN project_location ON project.job_no=project_location.job_no
INNER JOIN location ON project_location.name=location.name ";

$sql .= $where.$orderBy;

$result = mysqli_query($link,$sql);

if ($result !== 0) {

    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $name = htmlspecialchars($row['name'],ENT_QUOTES);
        $address = htmlspecialchars($row['address'],ENT_QUOTES);
        $city = htmlspecialchars($row['city'],ENT_QUOTES);
        $county = htmlspecialchars($row['county'],ENT_QUOTES);
        $state = htmlspecialchars($row['state'],ENT_QUOTES);
        $zip = htmlspecialchars($row['zip'],ENT_QUOTES);
        $country = htmlspecialchars($row['country'],ENT_QUOTES);
        $description = htmlspecialchars($row['description'],ENT_QUOTES);

            echo '<div title="Location: '.$name.'" style="padding:10px">';
            if ($address != '') {echo '<strong>Address: </strong> '.$address.'<br><br>';}
            if ($city != '') {echo '<strong>City: </strong> '.$city.'<br><br>';}
            if ($state != '') {echo '<strong>State: </strong> '.$state.'<br><br>';}
            if ($zip != '') {echo '<strong>Zip: </strong> '.$zip.'<br><br>';}
            if ($country != '') {echo '<strong>Country: </strong> '.$country.'<br><br>';}
            if ($county != '') {echo '<strong>County: </strong> '.$county.'<br><br>';}
            if ($description != '') {echo '<strong>Description: </strong>
            '.$description.'<br><br>';}
            echo '</div>';
    }
```

```
}

//Budget & Schedule accordion tab
$sql = "SELECT project.org_budget, project.fin_budget, project.budget_exp,
project.schedule_exp FROM project ";

$sql .= $where.$orderBy;

$result = mysqli_query($link,$sql);

if ($result !== 0) {

    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $org_budget = htmlspecialchars($row['org_budget'],ENT_QUOTES);
        $fin_budget = htmlspecialchars($row['fin_budget'],ENT_QUOTES);
        $budget_exp = htmlspecialchars($row['budget_exp'],ENT_QUOTES);
        $schedule_exp = htmlspecialchars($row['schedule_exp'],ENT_QUOTES);

        if ($org_budget != '' || $fin_budget != '' || $budget_exp != '' ||
        $schedule_exp != '') {
            if ($schedule_exp != '') {
                echo '<div title="Budget & Schedule" style="padding:10px;">';
            } else echo '<div title="Budget" style="padding:10px;">';
                if ($org_budget != '') {
                    echo '<strong>Original Budget: </strong> '.$org_budget.'
                    <p></p>';
                }
                if ($fin_budget != '') {
                    echo '<strong>Final Budget: </strong> '.$fin_budget.' <p></p>';
                }
                if ($budget_exp != '') {
                    echo '<strong>Explanation of Difference in Budget: </strong>
                    '.$budget_exp.' <p></p>';
                }
                if ($schedule_exp != '') {
                    echo '<strong>Completion on Schedule? If no, Why?: </strong>
                    '.$schedule_exp.'';
                }
            echo '</div>';
        }
    }
}

//Descriptions accordion tab
$sql = "SELECT project.deliverables, project.proj_desc_clos, project.work_desc_crea,
project.add_info_crea
FROM project ";

$sql .= $where.$orderBy;

$result = mysqli_query($link,$sql);
```

```

if ($result !== 0) {

    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $deliverables = htmlspecialchars($row['deliverables'],ENT_QUOTES);
        $proj_desc_clos = htmlspecialchars($row['proj_desc_clos'],ENT_QUOTES);
        $work_desc_crea = htmlspecialchars($row['work_desc_crea'],ENT_QUOTES);
        $add_info_crea = htmlspecialchars($row['add_info_crea'],ENT_QUOTES);

        if ($deliverables != '' || $proj_desc_clos != '' || $work_desc_crea != '' ||
        $add_info_crea != '') {
            echo '<div title="Descriptions" style="padding:10px;">';
                if ($deliverables != '') {
                    echo '<strong>Deliverables: </strong> '.$deliverables.' <p></p>';
                }
                if ($proj_desc_clos != '') {
                    echo '<strong>Project Description at Job Closure: </strong>
                    '.$proj_desc_clos.' <p></p>';
                }
                if ($work_desc_crea != '') {
                    echo '<strong>Work Description at Job Creation: </strong>
                    '.$work_desc_crea.' <p></p>';
                }
                if ($add_info_crea != '') {
                    echo '<strong>Additional Info. at Job Creation: </strong>
                    '.$add_info_crea.'';
                }
            echo '</div>';
        }
    }
}

//Dates accordion tab
$sql = "SELECT dates.event, dates.event_date FROM project
INNER JOIN dates ON project.job_no=dates.job_no ";

$sql .= $where.$orderBy;

$result = mysqli_query($link,$sql);

if (mysqli_num_rows($result) > 0) {

    $num_results = mysqli_num_rows($result);

        echo '<div title="Dates" style="padding:10px;">';
            echo '<table id="date" class="easyui-datagrid"
            style="width:352px;height:auto;">';
                echo '<thead>';
                    echo '<tr>';
                        echo '<th field="event" width="250">Event</th>';
                        echo '<th field="date" width="100">Date</th>';
                        echo '</tr>';
                    echo '</thead>';
        for ($i=0;$i<$num_results;$i++) {
            $row = mysqli_fetch_array($result);
            $event = htmlspecialchars($row['event'],ENT_QUOTES);
            $event_date = htmlspecialchars($row['event_date'],ENT_QUOTES);
                    echo '<tbody>';
                        echo '<tr>';
                            echo '<td>'.$event.'</td>';
                            echo '<td>'.$event_date.'</td>';
                        echo '</tr>';
        }
                    echo '</tbody>';
                echo '</table>';
            echo '</div>';

    }

    //Components accordion tab
    $sql = "SELECT component.component, component.manufacturer, component.component_desc,
    component.lessons
    FROM project INNER JOIN component ON project.job_no=component.job_no ";

    $sql .= $where.$orderBy2;

    $result = mysqli_query($link,$sql);

    if (mysqli_num_rows($result) > 0) {

        $num_results = mysqli_num_rows($result);

            echo '<div title="Components" style="padding:10px;">';

        for ($i=0;$i<$num_results;$i++) {
            $row = mysqli_fetch_array($result);
            $component = htmlspecialchars($row['component'],ENT_QUOTES);
            $manufacturer = htmlspecialchars($row['manufacturer'],ENT_QUOTES);
            $component_desc = htmlspecialchars($row['component_desc'],ENT_QUOTES);
            $lessons = htmlspecialchars($row['lessons'],ENT_QUOTES);

                echo '<strong>'.$component.'</strong> <p></p>';
                if ($manufacturer != '') {
                    echo '&nbsp;&nbsp;&nbsp;Manufacturer: '.$manufacturer.' <p></p>';
                }
                if ($component_desc != '') {
                    echo '&nbsp;&nbsp;&nbsp;Description: '.$component_desc.' <p></p>';
                }
                if ($lessons != '') {
                    echo '&nbsp;&nbsp;&nbsp;Lessons Learned: '.$lessons.' <p></p>';
                }
                $sql = "SELECT link.link, link.file_title FROM link WHERE
                link.component2='$component'";

                $result2 = mysqli_query($link,$sql);
```

```php

            if (mysqli_num_rows($result2) > 0) {

                $num_results2 = mysqli_num_rows($result2);
                for ($j=0;$j<$num_results2;$j++) {
                    $row2 = mysqli_fetch_array($result2);
                    $link3 = htmlspecialchars($row2['link'],ENT_QUOTES);
                    $file_title = htmlspecialchars($row2['file_title'],ENT_QUOTES);
                    if ($link3 != '') {
                        echo '&nbsp;&nbsp;&nbsp;<a href='.$link3.' target="_blank">'.$file_title.'</a><p></p>';
                    }
                }
            }
    }
        echo '</div>';
}

//Links accordion tab
$sql = "SELECT link.component2, link.file_title, link.link, link.file_desc
FROM project INNER JOIN link ON project.job_no=link.job_no ";

$sql .= $where.$orderBy3;

$result = mysqli_query($link,$sql);

if (mysqli_num_rows($result) > 0) {

    $num_results = mysqli_num_rows($result);

        echo '<div title="Links" style="padding:10px;">';

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $component2 = htmlspecialchars($row['component2'],ENT_QUOTES);
        $file_title = htmlspecialchars($row['file_title'],ENT_QUOTES);
        $link2 = htmlspecialchars($row['link'],ENT_QUOTES);
        $file_desc = htmlspecialchars($row['file_desc'],ENT_QUOTES);
            echo '<strong>'.$file_title.'</strong> <p></p>';
            echo '&nbsp;&nbsp;&nbsp;Link: <a href='.$link2.' target="_blank">'.$link2.'</a> <p></p>';
            if ($component != '') {
                echo '&nbsp;&nbsp;&nbsp;Component: '.$component2.' <p></p>';
            }
            if ($file_desc != '') {
                echo '&nbsp;&nbsp;&nbsp;Description: '.$file_desc.' <p></p>';
            }
    }
        echo '</div>';
}
    echo '</div>';

mysqli_close($link);
?>
```

Appendix F

PHP Code – Combo Box Lists

Job Number

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT job_no, job_name FROM project GROUP BY job_no;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $job_no = $row['job_no'];
        $job_name = array($job_no,' - ',$row['job_name']);
        $output = implode('',$job_name);
        $items .= '{"id":"'.$job_no.'","text":"'.$output.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Project Manager

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT CONCAT(project.pm_last, ', ', project.pm_first) AS name FROM project
GROUP BY project.pm_last, project.pm_first;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $name = $row['name'];
        $items .= '{"id":"'.$name.'","text":"'.$name.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Client

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT client_name FROM project GROUP BY client_name;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $client_name = $row['client_name'];
        $items .= '{"id":"'.$client_name.'","text":"'.$client_name.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Client Contact

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT CONCAT(project.contact_last, ', ', project.contact_first) AS contact
FROM project GROUP BY project.contact_last, project.contact_first;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $contact = $row['contact'];
        $items .= '{"id":"'.$contact.'","text":"'.$contact.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Prime Contractor

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT prime_name FROM project GROUP BY prime_name;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $prime_name = $row['prime_name'];
        $items .= '{"id":"'.$prime_name.'","text":"'.$prime_name.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

ESG Company

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT esg_comp FROM project GROUP BY esg_comp;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $esg_comp = $row['esg_comp'];
        $items .= '{"id":"'.$esg_comp.'","text":"'.$esg_comp.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Location Name

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT name FROM location GROUP BY name;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $name = $row['name'];
        $items .= '{"id":"'.$name.'","text":"'.$name.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Type of Work

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT work_type FROM type_of_work GROUP BY work_type;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $work_type = $row['work_type'];
        $items .= '{"id":"'.$work_type.'","text":"'.$work_type.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

State

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT state FROM location WHERE state <> '' GROUP BY state;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);
    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $state = $row['state'];
        $items .= '{"id":"'.$state.'","text":"'.$state.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';

}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

City

```php
<?php
$state = $_REQUEST["state"];
$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT city FROM location WHERE state = '" .$state. "' AND city <> '' GROUP BY city;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);
    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $city = $row['city'];
        $items .= '{"id":"'.$city.'","text":"'.$city.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Contract Type

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT contract_type FROM project GROUP BY contract_type;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $contract_type = $row['contract_type'];
        $items .= '{"id":"'.$contract_type.'","text":"'.$contract_type.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

State Job

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$labels = array(
        "Y" => "Yes",
        "N" => "No"
);

$sql = "SELECT state_job FROM project WHERE state_job is not null GROUP BY state_job;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $state_job = $row['state_job'];
        $items .= '{"id":"'.$state_job.'","text":"'.$labels[$state_job].'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Federal Job

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$labels = array(
        "Y" => "Yes",
        "N" => "No"
);

$sql = "SELECT fed_job FROM project WHERE fed_job is not null GROUP BY fed_job;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $fed_job = $row['fed_job'];
        $items .= '{"id":"'.$fed_job.'","text":"'.$labels[$fed_job].'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Project Component

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT component FROM component GROUP BY component;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $component = $row['component'];
        $items .= '{"id":"'.$component.'","text":"'.$component.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

Manufacturer

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT manufacturer FROM component GROUP BY manufacturer;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $manufacturer = $row['manufacturer'];
        $items .= '{"id":"'.$manufacturer.'","text":"'.$manufacturer.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```

File Title

```php
<?php

$link = mysqli_connect("localhost","user","password","esg_projects");

$sql = "SELECT file_title FROM link GROUP BY file_title;";

$result = mysqli_query($link,$sql);
if ($result != "0") {
    echo '[';
    $items = "";
    $num_results = mysqli_num_rows($result);

    for ($i=0;$i<$num_results;$i++) {
        $row = mysqli_fetch_array($result);
        $file_title = $row['file_title'];
        $items .= '{"id":"'.$file_title.'","text":"'.$file_title.'"}, ';
    }
    $items = substr($items, 0, -2);
    echo $items;
    echo ']';
}
else {
    echo 'Problem with query!';
}

mysqli_close($link);

?>
```