Service Learning Practicum: LEAP Data Management and Visualization System

James Ashe
University of Virginia
jga9je at virginia.edu

Jonathon Blonchek
University of Virginia
jrb2yk at virginia.edu

Vinay Dandekar
University of Virginia
vvd5gc at virginia.edu

Cameron Nye
University of Virginia
cpn2pf at virginia.edu

Renee Seaman
University of Virginia
res6tq at virginia.edu

Eric Tsai
University of Virginia
ekt7be at virginia.edu

ABSTRACT

The capstone team, consisting of six computer science undergraduate students at the University of Virginia (UVA), was connected with the Local Energy Alliance Program through the yearlong Service Learning Practicum (SLP) capstone course. The goal of the Service Learning Practicum is to give students experience developing large software projects and interacting with clients in the local Charlottesville community. The Local Energy Alliance Program (LEAP; http://www.leap-va.org) helps increase the energy efficiency of residential buildings by installing and monitoring energy efficient technologies. LEAP's efforts save homeowners money, create jobs in the Charlottesville community, and reduce environmental harm. Before the system was created, LEAP was only able to analyze the data from 30 homes out of the thousands LEAP has completed retrofits on. To increase LEAP's local influence, the capstone team created a website and database using Ruby on Rails and MySQL to help LEAP store and analyze the energy usage data of buildings. The database management system compiles data from utility companies with a home and project database. Energy data is collected prior to and after implementing the energy-efficient technologies in the buildings, so the effectiveness of LEAP's work can be measured. The website interacts with the database to support report generation, data visualization, and extraction of statistics. This automation of the analysis will increase LEAP's productiveness because they will no longer have to do the reporting and analyzing by hand with spreadsheets. The final system allows LEAP to streamline the process of analyzing the total energy saved per year, and project how much money homeowners will save on their bills by implementing energy efficient solutions.

1. INTRODUCTION

The Local Energy Alliance Program (LEAP; http://www.leap--va.org/) helps make residential and business buildings more energy efficient by completing energy evaluations, scheduling retrofits with approved contractors, and then monitoring the energy savings. Retrofits are unique for each building, based on the building's energy evaluation, but include updating installed measures such as windows, exterior doors, attic insulation, and programmable thermostats.

Before UVA's SLP students developed a data management and visualization system for LEAP, LEAP used Microsoft Excel to manage their clients' information such as property owner details, energy consumption, and retrofit records. In order to complete full analysis of a property, they had to manually consolidate their data from multiple excel files. Once the data for one property was compiled and formatted correctly, the data was uploaded to PRISM, an external software system used for weather normalization.

That method of completing data management and analysis takes a long time, and is not guaranteed to include all available data. Therefore, LEAP was only able to completely process and analyze data from 30 buildings, leaving a large majority of data unanalyzed.

LEAP needed a unique tool that could automate the consolidation and storing of their data accurately and quickly. The system required a single database to hold all of LEAP's data, and a website to support data visualization which would allow LEAP to upload and manage data from varying sources.

One of the biggest reasons the client could not use an existing system is that their data is not uniform and is often not sanitized due to being imported from different sources. Also, the PRISM software requires data to be in a specific format that no existing tool provides.

The capstone team successfully made a customized solution to the problems LEAP was facing—the LEAP Data Management and Visualization System. This system allows the organization to effectively analyze large quantities of data by automating and streamlining the processes LEAP previously had to do by hand with Excel. This will allow LEAP to save time and expand their impact on the Charlottesville community by improving the energy efficiency of more homes in Virginia.

The system was developed by six students in the University of Virginia's Computer Science Engineering department, under the Service Learning Practicum capstone course. This course was designed to give UVA undergraduate students experience with developing a real software product with a small team using an agile development strategy. The team worked together for the yearlong course with advice from a mentor to develop software development skills and to produce a working system for a client.

2. BACKGROUND

This system was designed to speed up the analysis workflow of LEAP and uses the popular Ruby on Rails framework and a MySQL database to store and retrieve data. It streamlines the process by providing functionality to manipulate data regarding properties, energy measurements, and installed measures.

LEAP provides cheap housing evaluations to property owners. Based on the housing evaluation, suggested retrofits are installed with an approved contractor. When all of the retrofits to a property have been completed, a date called the "test out date" is assigned to that property. This date allows LEAP to compare the energy consumption data before and after completing retrofits.

LEAP gathers data for their client properties by requesting data from utility providers such as Dominion and Washington Gas. Customers are also asked to provide billing statements so LEAP can obtain the meter reading dates of the monthly energy consumption. The utility providers give LEAP data in the format of Excel sheets and LEAP has to import the data from the Excel into their database. If the Excel sheet contains data on account number, date read, consumption, and days used, then LEAP can upload the Excel sheet into the system developed by the capstone team. The system will then upload all the data into the database which saves LEAP the trouble of manually inputting data.

The system is designed for LEAP's internal use only. The users are all administrators of the system. To keep the system internal, only existing users can create new user accounts. Therefore, unless a current user account is leaked, anyone without an account will be unable to use the website. Because the system is online, LEAP employees can manage their data from any location with an internet connection.

3. RELATED WORK

Multiple generic software packages exist for utility data management from vendors such as Oracle and Cisco Systems. These packages are targeted at utility companies rather than smaller organizations such as LEAP. As such, they are far too complex and expensive for a nonprofit organization that has no technical employees.

In the past, LEAP used Excel spreadsheets to manage their data. This took unreasonable amounts of time to produce a single set of analyzable data. Also, the data within the spreadsheets was not consistent and data for one property was spread across multiple spreadsheets.

A relational database technology (such as SQL) could be used as a standalone solution, but that would require many different complicated queries. Our client does not have the skillset to work directly with relational database queries, so our system includes a convenient frontend for the database. Compared to the commercially-available solutions, ours functions as a lightweight alternative with a smaller set of features focused towards LEAP's needs.

4. SYSTEM DESIGN

The system is built off the Ruby on Rails MVC framework and utilizes MySQL for database queries. Because the system is meant to manage data, a majority of the system is comprised of several models that represent LEAP's data as well as controllers to handle manipulation of the data.

4.1 Entities

4.1.1 Properties

Properties are the key units of interest that LEAP analyzes and manages. Therefore, the system has a well-defined property model that contains data about the specific property such as address and owner name. The system provides basic CRUD operations as well as functionality for searching for, filtering, and sorting properties. Each property was previously assigned a unique customer ID by LEAP which the system currently uses in conjunction with a separate integer primary identifier. The system has separately defined recordings and installed measures models because a property has many recordings and installed measures.

4.1.2 Installed Measures

Installed measures are the energy-saving installations that a house or property has had completed on it. For example, duct insulation, wall insulation and air sealing would be considered installed measures. A simple model is defined for installed measures to allow the addition of new types of installations or editing existing installed measure types. As specified by LEAP in the requirements, existing installed measures cannot be deleted. The only field required for the model is the name of the installed measure such as "heat pump" or "duct insulation."

Multiple installed measures can be associated with a property. Installed measures are associated with specific properties through a relational table in the database. When an installed measure is associated with a property, there is a text field available where the user can specify any additional information on the installed measure for that specific property such as contractor or number of new elements. This comments field is not searchable in the system. Yet the specific installed measure types can be filtered on so leap can compile a list of all properties with the same retrofits to complete a more focused statistical analysis.

4.1.3 Recordings

A recording is representative of a month's worth of data on energy consumption. Recordings are associated with a utility type and account number. The account number is the property owner's account number from their respective energy company, e.g., Dominion. The property has many recordings based on the owner's account number. The recordings are accessed through a system record lookup table because a property can have multiple account numbers. The record lookup table ties properties to multiple account numbers and the account numbers are related to multiple recordings.

4.1.4 Utility Types

The utility type defines the energy type of the recording. A utility type model is defined for the each utility type in the system. Each utility type has a primary identifier, type name, and unit associated with it. The current utility types include electricity, natural gas, and water, but the relational database was created to give LEAP the option of adding additional utility types in the future.

4.2 System Features

4.2.1 Finding Gaps in Data

One of the most important features of the system is identifying and compiling a list of all missing data that is required for analysis. The database is held to a rule that specifies a given energy account number can only have one recording associated with it in a given month. This is a standard enforced on data entry, rather than in the actual database schema. For a given property, the customer requires data in a 12 month radius around the test out date. The interesting problem arises when the system needs to decide which month a given utility reading applies to. This is difficult because meter readings can occur on any day of the month. Therefore, the architects of the system decided that a reading in the first 15 days of a month applied to the previous month's data while a reading occurring in the latter half of the month applied to the same month as the reading. This design decision allowed the gaps report to have an easy way to assign data to where it "exists" and to show a gap where it did not.

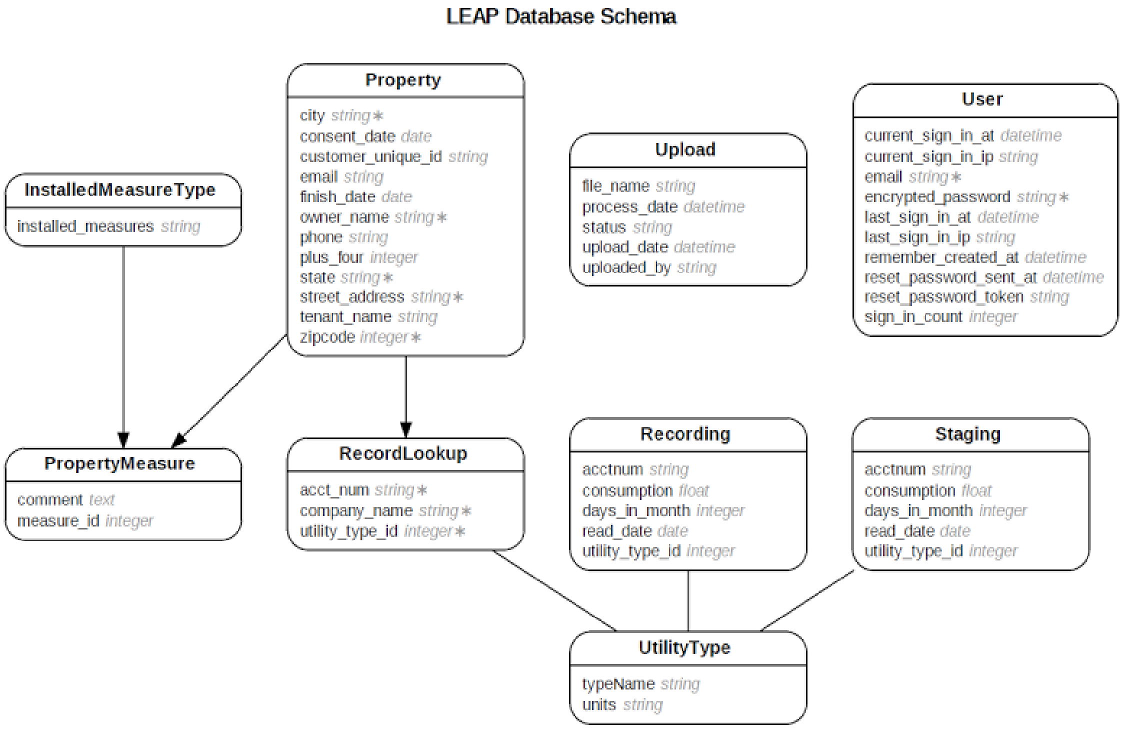

Figure 1. Database schema for LEAP's Data Management System.

4.2.2 Filtering

The system can filter properties based on criteria that includes zip code, a range of test out dates, and installed measures. Once the database entries corresponding to the filter selections is found, the list of properties is displayed to the user and recordings for those properties can be exported in a PRISM-ready format. This exporting is done using a specific SQL query, as opposed to the Rails ActiveRecord interface, because the query is too complicated for the built in Rails framework.

4.2.3 Exporting Data

The system can export reports based on SQL views that are designed specifically for the customer's data request or analysis needs. Many of the customer's data requests require a lot of specific logic, needing specific queries in the database which would be difficult to design using the Rails framework. The system wraps these rendered views any time a request is made as a report. Any analysis or request report can be accessed by the customer through the simple tabular sheet on the website.

4.2.4 Search

In order to allow LEAP the easiest method of finding energy consumption for specific properties, two search bars were implemented-- one in the navigational bar and another on the home page. Searching for a property is a major feature of the system which gives LEAP prompt access to specific properties. By placing the search bar with the navigational bar, the system improves LEAP's accessibility to properties by allowing users to search from anywhere in the system. The search itself specifically queries for both the property owner's name and the address of the property. Therefore, LEAP can search for different properties that may be under the same owner name. To make properties even more accessible, the search bar uses Twitter's Typeahead JavaScript library to implement autocomplete. The autocomplete includes functionality for searching by owner name or address. The autocomplete dropdown of hints is based off a list of all properties and their owner names that can be either hard-coded or passed to the page. The list of all properties is accessed during every page load because the searching functionality is on all pages. To prevent the list from being publicly visible in each page source, the list is JSON serialized and given through a prefetch URL.

4.2.5 Users

Security is a main concern for the client because LEAP handles property owners' confidential data including utility account numbers and energy usage. The Devise gem for Ruby on Rails was used to create the user login functionality of the system and make the system secure. Devise was able to be customized to fit LEAP's specific needs by allowing only existing users to create new users and delete existing users. To accomplish the customization, the Devise views were generated and altered. Then new routes and controllers were created to support the additional functionality of adding and deleting users. The Devise controller was unable to be used because it only allows a new user to be registered when no one is logged in.

4.3 Design Decisions

LEAP owns an existing Amazon EC2 instance which already contains LEAP's data. Despite the existing database, a decision was made to design a new database specifically for this system. The existing database did not contain all of LEAP's data. Determining which LEAP data was or wasn't in the existing database was deemed harder than inputting data from scratch. After a quick database analysis, the existing database was determined to be too complex. A simpler database could be designed after LEAP's data was reorganized and cleaned up.

LEAP generates a customer unique ID using their internal management systems. The first decision made was whether or not to use this ID as a primary key for a given property in the database. Based on the data we received for input into our system, where we observed a lack of consistency and presence of the unique ID, so we designed the system to generate its own unique primary key attribute to maintain consistency across all data new and old.

One of the more valuable decisions made in the database design was the separation of account information from a specific property. This preserve normality in the database, as a given property can have more than just two account numbers for a given type of energy. As can be seen in Figure 1, the database (see Figure 1) contains a table called RecordLookups. This table is responsible for maintaining all relationships between properties and utility data points. The table joins the primary key of property with an account number in the recordings table for easy access, lookup, and expandability.

LEAP provided data in spreadsheets that contained information on properties including an owner name as well as a tenant name. A major issue encountered was owner name and tenant name would differ for certain properties. This caused some confusion as to what name was associated with the utility account numbers. Sometimes the utility account was tied to the owner name and sometimes it was tied to the tenant name. Too many exceptions existed if the system were to try and accept data with both owner name and tenant name. As a result, the non-sanitized data was cleaned by adjusting all data such that owner name was the only name associated with the account numbers.

5. PROCEDURE

The system is designed to be an all-inclusive solution for the data processing needs of the customer. The system should only be used by a select number of people. LEAP is a small organization with an even smaller number of people assigned to doing the energy data analysis. These few people will be the only users with access to the data and analysis tools. The user will log in and the system presents a variety of options, all specified by the customer in the initial requirements document. The system has two main purposes: to aggregate data for a given property, and allow for export of data to external tools for further analysis. The dashboard of the system presents a search bar where the customer can search for a given property. A property report is generated for the customer where they are given various options:

Gaps Report: A display of the data points acquired and needed for a full 24-month analysis.

Raw Data: Displays raw data in the range, used primarily in case of system error.

Edit Property Information: Allows editing all database information associated with the property. This includes account numbers and test out dates.

Generate PRISM Report: Export this specific properties data for input into the PRISM program.

The user may also generate a variety of reports for both analysis and requests. Analysis reports display properties with a full utility data set that are ready to be exported from the system so statistical analysis can be completed on them. Request reports provide the information necessary for LEAP to send to energy providers to collect data. The user can also filter properties based on a variety of metrics including installed measures, test out date range, and zip code.

6. RESULTS

Prior to this project, LEAP Virginia's data management system was extremely unorganized and not well coordinated. The client simply could not tell if they had the data they needed, and were not able to locate the appropriate data when it did exist. All of the data was originally stored in spreadsheets, in many different formats, and across a large number of computers. The benefits of having one central location for all the information are substantial. The new system designed in this course will not allow redundancy or inconsistencies in data which were common before. This system has become the one location for all of LEAP's data needs. It is intended to carry them into the future and allow for the organization to grow and expand to new territory and new revenue opportunities.

Without the new system, the client was very slow in processing the data needed for their work. Assuming they have all the data pre-compiled, it would take them roughly 10 to 15 minutes to prepare one report for the PRISM program. Using the new system, the client is able to accomplish this same task in about 1 to 2 minutes for not just one, but any amount of properties. Similarly, it took them 15 to 30 minutes on average to find the information on one property without using our system. With our system, it should only take on average 1 to 2 minutes to accomplish the same task. The new system offers a variety of analysis reports, making data collection much quicker and easier. When requesting meter reading data from a utility company, in the past, those request were generated by hand. With the new system, that process is completely automated to request the date ranges required for accurate analysis. The new system has optimized and streamlined virtually every process that data analysts at LEAP had to previously perform manually.

7. CONCLUSIONS

The capstone team designed and developed a data management and visualization system which automates and streamlines the workflow of the nonprofit organization, the Local Energy Alliance Program.

All of LEAP's data was consolidated into one database. The corresponding website facilitates the easy finding and analyzing of LEAP's data. This system is a drastic improvement over LEAP's previous system of housing data in multiple Excel files because it decreases the amount of time it takes LEAP to analyze a property's utility data, automates formatting reports for PRISM, and compiles a list of missing data into a utility request format. It also helps ensure the data is more accurate because the system makes the data uniform and cuts down on human error.

The functionality the team developed, including the Gaps Report, Utility Requests, and Analysis, significantly decreases the time it takes LEAP to complete their tasks. The LEAP Data Management and Visualization System also supports functionality beyond what LEAP was able to originally accomplish. The filtering functionality now allows LEAP to find all properties corresponding to specified parameters and complete analysis tailored to those specifications.

No existing programs would have been able to meet the unique requirements of LEAP. The system developed by the capstone team was designed specifically to interact with the other programs LEAP uses. An example of this is PRISM-- all lists of properties a user could generate are able to be exported from the new system in a PRISM ready format.

In addition to the project benefiting LEAP, the project benefitted the capstone team by exposing the students to a long term software project where they had the opportunity to utilize and develop their software development skills through interacting with the client, developing the necessary documentation, designing a usable system, and testing to ensure the system meets the client's needs.

8. FUTURE WORK

While the system fixes the client's problems, meets LEAP's needs, and accomplishes the goals defined in the requirements, there are more features that could be added to the system to extend its use. Extending functionality of the system could improve its usability and usefulness. Basic functionality that could be updated includes improving the data handling on upload. Due to the nature of the mismatched data, handling all of the various cases involved in the data upload goes beyond the initial scope of the project. Upload could include support for homeowner and housing data. Also, better visualization alerting the user to problems in their data such as conflicts with existing data could be added. More extended functionality would involve better data visualization and statistical analysis in general, such as graphs to chart the data and allow the user to easily see how their data changes over time. Comparing properties to see how their data compares might also prove useful to the client. Due to time constraints, these features could not be implemented in the current system, but they are ideas for future implementation.

9. ACKNOWLEDGMENTS

A special thank you to Jon Proffit and the LEAP organization for their support and willingness to work with us throughout this learning experience. Though they have greatly benefited from this project, we have gained equal if not more. It was a true pleasure working with LEAP. Also, a sincere thanks to John Feminella for his guidance as the group's mentor throughout the process. His guidance was very important at key times throughout development, and he significantly aided our development as software developers.