Civera – Revised SOW	
Contact	Adam Friedman Founder Civera Software
Organization	Civera Software
Organization Description	Civera Software provides technology to build a better democracy. Our customers typically range from state governments to public-interest organizations. Our passion is to liberate civic data and make it easy for everyday citizens to access and use it.
Project Type	Data Science
Project Description	This project involves standardizing data fields for scraped housing data from the Massachusetts court system. The data was collected from masscourts.org into a more user-friendly searchable database at masscourtsplus.org, the court dockets include fields such as: Parties involved, court name, date filed, last action date, and court ID number. The dockets can be further elaborated to examine a timeline of case activity of which there are 100-300 type of case activity events. The students will process the raw data from the client and use existing PHP code using regex to standardize the fields and convert that into a more accurate and scalable Python module. This will be done by both replicating the regex logic into Python as well as entity recognition, and machine learning methods to group together variations. The standardization of the fields will create more searchable features in a future iteration of this database and allow analytics dashboards.

Data Sets	**Data Sets:** Client's raw data for MA housing court dockets. Our datasets include four tables we access through MySQL with millions of records each. Initially, we'll focus on the first table and its description field. wp_courtdocs_cdocs_case_action_index wp_courtdocs_cdocs_case_meta_data_index wp_courtdocs_cdocs_party_assignment_index wp_courtdocs_cdocs_party_index

Approach	**Revised Statement of Work (SOW)** After meeting with Adam Friedman on March 4, we agreed that since steps 3 and 4 (outlined below) are heavily dependent on doing a thorough task with step 2, step 2 is our first priority. Adam's primary purpose is to normalize the database. To infer actor and action from the descriptive fields they mine from courthouse records. He's also like to update the regex code written in PHP. But normalizing what they've already collected takes priority and will be an ongoing effort over the course of multiple semesters. Since latter steps, like inferring from the normalized data, are heavily dependent on the assignment of records *(like actor and action)* we agreed to focus on the assignment of these labels initially. If we can normalize actor and action, we can start to make inferences regarding other case actions and other meta data about the case. We agreed that the initial focus should be on pulling rows from the **case_action_index** table, use spaCy for semantic analysis of the description field to determine the values for the **Actor** and **Action** fields. Adam has normalized around 30%, the remainder are missing these critical fields. We've been given access to his legacy code on GitHub, the databases on MySQL and write access to a database on GitHub for our results. **Step One:** Update the Python scraper code provided by the client and update the housing court docket data for the most recent years. **Step Two:** Convert PHP regex code into Python regex code and explore how many of the features are successfully reduced into common categories. ● Depending on the level of success, the students may have to tweak the regex patterns or introduce entity recognition to increase accuracy. **Step Three:** Improve the regex feature normalization with entity recognition that can quickly identify common words and combine this with regex. **Step Four:** Further improve the entity recognition by perhaps incorporating clustering algorithms such as K-means clustering to normalize the features. **Step Five:** After feature normalization, create analysis on which types of features (by case activity, party type, etc) affect case duration, and the damages paid. Regression analysis can be used here.

Additional Information	**Tools and Methods** *Data processing:* Pandas, NumPy and spaCy will be used initially to extract text from the SQL databases and populate the actor and action fields. *ML & Entity Recognition:* Scikit-learn, spaCy and nltk to use entity recognition to speed up regex pattern recognition.