Buy Back Your Vote

Identification:

Name: Buy Back Your Vote

Manager: Tim Bauman (tbauman@princeton.edu)

Member names:

- Margaret Fortney (mfortney@princeton.edu)
- Matthew Dolan (mdolan@princeton.edu)
- Christopher Kelly (cekelly@princeton.edu)
- Nathan Keyes (nkeyes@princeton.edu)
- Tim Bauman (tbauman@princeton.edu)

http://www.cs.princeton.edu/~tbauman/333.html

Overview:

For our project, we will create both a web application and iPhone app called WalletVoter. WalletVoter lets users look up companies' political contributions, including the specific politicians they support and bills they have lobbied for. In recent years, political donation levels have skyrocketed. With the Citizens United vs. the Federal Elections Commission Supreme Court case ruling that corporations and trade unions may make unlimited political expenditures, this is only likely to increase. Because corporations and donors sometimes use complicated structures like PACs to send politicians so-called "soft money," it can be hard to track who gets funded from where. The goal of this application is to allow politically-minded consumers to learn more about the political stances of the companies who produce the products they buy. They will obtain this information by searching for products by barcode or by manufacturer name through their web browsers and their mobile devises.

This website and application is inspired by existing sites such as the Boycott SOPA Android app, Good Guide, and OpenSecrets. Boycott SOPA reports whether or not a product's manufacturer supports SOPA, Good Guide searches for environmental and health facts about products, and OpenSecrets tracks contributions to political campaigns. Our project will draw on the same data sources as these previous websites, but we will approach the display of this data in a different way. We are focusing on companies' political campaign contributions made by corporations in the current election. To our knowledge, no other app or website has the same focus on the interactions between companies that manufacture products and politicians.

On the website, a user will be able to see a list of top contributors and top searched companies. There will also be a search bar that will allow users to find companies by name. Each company will have a profile page detailing its political involvement. The iPhone app will be similar: it will allow users to search for a company either by name or by a product's barcode. Both the iPhone app and the website will direct users to a page that lists the donations and positions of the company they searched for.

Functionality:

The main focus of our application is to connect the user with the data on campaign contributions. Below is a list of the various ways we imagine users can reach this final data.

There is only one user role for the website and the iPhone, so there are only a few user cases we will outline, and many of them end the same way. Where a previous page or feature was described before, we have not repeated that information.

Use Cases:

User - browsing website:

User loads front page in desktop web browser. The front page has general information about our project. This includes information about what the project is and where the data comes from. Beyond this basic information, the front page also contains a couple of interesting lists that might direct the user to interesting content. Among these lists is the highest contributors in total dollar amounts, companies with most page views, and possibly top contributes by candidate. The user clicks on the top contributor (say, Coca-Cola). This brings him to a web page for Coca-Cola that displays all of Coca-Cola's campaign contributions through various subsidiaries, PACs, political parties, and lobbyists, broken down by party and geographic region. It also displays a list of bills that Coca-Cola is known to support. The user returns to the front page and clicks on another company.

User - searching for company on website:

User sees the front page (described above). At the top of the front page, there is a search box. The user clicks on the search box and enters a company name (say, "coca cola"). As the user begins to type, an auto-complete box drops down to suggest possible company names. The user presses enter to search. Since “coca cola” matches Coca-Cola very closely, the user gets redirected directly to the Coca-Cola profile. The user views the Coca-Cola profile as described above.

If the user had entered a more ambiguous term, the user would see a page that displays top matches as available. It will show them in order of relevance and show a brief summary of Coca-Cola’s contributions.

User - searching for company by barcode on iPhone:

The user sees the front page of the iPhone app. There is a search by barcode button, a search box for searching by name, and a “Go” button for the search by name box. The user taps the barcode button. This brings up a barcode scanning screen. The user hovers over a barcode until it is recognized. This brings the user to the company’s profile page. This profile page is identical to the one on the website, although optimized for a smaller screen.

User - searching for company by name on iPhone:

The user sees the front page of the iPhone app. The user taps the search by name box.
The user enters “coca cola” using the on-screen keyboard. The user presses “Search.” The company profile page appears.

Design:

The website will be written using the Django platform. As a web application, the pages will be built from HTML (generated through Django’s template system). Because our data is long term and unlikely to change in real time, AJAX will likely not be necessary. If it is, we will use jQuery to query and deliver content. Much of the display features will be created using the Google Charts API. This API will require using Java Script to define interactive charts on the client end. Finally, the style will be defined with CSS sheets generated using Bootstrap, a open source style platform created by Twitter. Although Bootstrap give a great deal of functionality out of the box, we will customize the style using Less programming to create the CSS sheets.

The data will be organized with a PostgreSQL table, although some of the information will be drawn from remote APIs provided by OpenSecrets, the Sunlight Foundation’s Influence Explorer, and MapLight. The database will most likely be generated and queried in Python using the Django framework. The search functionality will likely be run by Solr or Sphinx.

The website and database will both be hosted on Heroku, which will be relatively easy to set up with Django, Sphinx, and a PostgreSQL database.

The iPhone app will be written in Apple's Xcode iOS development environment using Objective-C.   We will use an open source barcode scanning library that is available for iOS. The iPhone app will query our server (using an HTTP request), which will return the data for the company's profile page. We will then display the data in the app, probably using HTML in a UIWebView.

Data organization:

The APIs we will use are available at http://data.influenceexplorer.com/api, http://maplight.org/apis/bill-positions, http://code.google.com/apis/shopping/search/v1/reference-overview.html.

Our goal is to map a given company to a political position, whether it is a party, a candidate, or a bill. The ways that money gets from companies to political positions are numerous: PACs, Industry interest groups, lobbying, direct donations to candidates, SuperPACs, donations through parent and subsidiary companies, and verbal support of candidates.

Most of this data is accessible through MapLight and InfluenceExplorer's APIs, but some of it will be stored in our database for ease of access. Currently, we are planning on storing a table of companies with the option of having subsidiary and parent companies so that we can search for companies easily and find their related companies. This way, we can display all of this related data together.

Company:

- id
- name
- location
- industry
- revenue
- parent (reference to company id)
- maplight_id (for searching MapLight's database)

Influence Explorer has lists of donations to PACs, campaigns, and so on. We will use this data to find who each company is donating to, and if the company is donating to an organization, we will continue to follow the money until it reaches a specific campaign.

We will also use the Google Product API on the iPhone to look up UPC codes (better known as bar codes). We can simply search it using UPC codes to find potential products.

The relevant fields that Influence Explorer returns are: parent_organization_name, organization_name, committee_party, committee_name, contributor_zipcode, contributor_employer, contributor_type, recipient_name, recipient_party

The relevant fields that MapLight returns are: disposition, measure, topic, catcode

The relevant fields that Google Products API returns are: items, title, brand

Milestones:

March 17
- Design document complete
- Server running
- Versioning (Git) running

March 24
- Database schema started
- Django project initialized
- iPhone app scans barcodes

March 31
- Website handles searching

April 7
- iPhone app sends requests to the server/server receives requests

April 13
- **Prototype**
- Website displays basic company page
- iPhone app displays company page from website

April 20
- Website & iPhone displays complete company page

April 27
- **Alpha test**
- Website displays list of top contributors

May 4
- **Beta Test**
- Clean up of major features and bug fixing

May 15
- **Dean's Date and submission deadline**

Risks and Open Issues:

Barcode Scanning:

One of the main open issues is to make sure that the open source bar code scanning software is capable of accurately picking up bar codes and connecting to a database that provides us with accurate information about product manufacturers. For example, we expect to use the ZBar Open Source barcode reader, but we have not yet been able to run and test the source code on an iOS device that has a camera, so there may be unexpected complications in getting that to work. Another possible barcode reader is RedLaser, but it has a very limited free trial period, and we may find ourselves needing to pay for a developer account.

Data Amalgamation:

We are looking to gather data from a couple of data sources and combine it in a way that makes it a form that can be easily searched. This include linking PAC connections back to the candidates that they support, connecting company donations from different sources and linking them back to a single company type. This type of problem we hope to approach it as we become more familiar with the data.

Project Timeline Document:

Our timeline document is available at http://www.cs.princeton.edu/~tbauman/Timeline.html