**ELECTION SIMULATION**

**TECHNOLOGY MANIFESTO**

1. **TECHNOLOGIES**

*Web hosting*

**Windows Azure**

Free for to use within the scope of class.

Has great industrial capacity.

Strong community

*Web framework*

**Django web framework**

Client requires that the system will be able to runs on multiple platforms. Django is written in Python which can run on almost all popular operating system (Windows and UNIX like systems). Also Great Bear decided to make a website, so users can access easily from any operating system without special technical support.

*Database*

**Comma separated values (CSV)**

**SQLite 3**

Great Bear decided to convert all files to a CVS and then to SQLite for future immigration to a different database. Because SQLite is compact lightweight, portable database so it has a lot of benefit during early stage of development, and learning. However, because of being lightweight, later on, SQLite will have a lot of performance issues, so SQLite is used to be replaced.

Using a CSV as a buffer will simplify the future immigration procedures. Future development will have to only care about converting from a standardized CSV format to a new DB format.

*Repository*

**Github**

An industrial tool

*IDE*

**Pycharms community edition**

**Eclipse with PyDev plugin**

These tools have good reputations and community support.

These tools have good version control features

*Test*

**Selenium**

Testing suite for testing the dynamically-generated values for the html files

1. **FILE STRUCTURE**
   * ElectionSimulation : All the code for the website
     + ElectionSimulation : Main code for the website
       - CampaignMaster : The views file and url patterns at campaign master level
       - Models : The all models for the sites
         * models.py : Have import statement to include all the models for the site
         * <Model names> : Model files
       - static : All the static files for the website
         * bootstrap : All the js, css, image files for Twitter Bootstrap
         * css : All extra css files for the sites
         * js : All extra css files for the sites
       - templates
         * CampaignMaster : All the layout and templates for Campaign master section

layout.html : The layout for campaign master’s pages

list.html : The content for campaign master’s listing page

* + - * + layout.html : Main layout for the entire site
        + User: All the layout and templates for User

layout.html : The layout for user page

list.html : The content for user listing page

add\_candidate\_effort.html: Page where user can specify the candidates' efforts for the simulation

piechart.html : Page where the graph of the simulation is showing

* + - * User: The views file and url patterns at User Level
    - Setup.py: It saves data into the respective models(user, campaign, etc), so that the models can be used to access the data later
    - Login: All the layout and templates to login

1. **MODELS DESIGN**

*Note: Remember to run “****python manage.py syncdb****” to create the DB file before running the server*

* + **Campaign**

ID : the id of a campaign – *This is made by the framework*

name : the name of the campaign

create\_date : The date of creation

candidates : List of all candidates and their party separated by new line character

voting\_system : The name of the voting system

formula : The foreign key to the formula

* + **Formula**

ID : the id of a formula– *This is made by the framework*

name : the name of the formula

* + **PoliticalParty**

ID : the id of a political party– *This is made by the framework*

name : the name of the political party

* + **Riding**

ID : the id of a riding– *This is made by the framework*

name : the name of the riding

* + **Ctract**

ctuid: the unique id for the census tract

age\_1: relative weight for the distribution of effort to the first age group

age\_2:relative weight for the distribution of effort to the second age group

age\_3:relative weight for the distribution of effort to the third age group

age\_4:relative weight for the distribution of effort to the fourth age group

age\_5:relative weight for the distribution of effort to the fifth age group

age\_6:relative weight for the distribution of effort to the sixth age group

age\_7:relative weight for the distribution of effort to the seventh age group

age\_8:relative weight for the distribution of effort to the eighth age group

age\_9:relative weight for the distribution of effort to the ninth age group

age\_10:relative weight for the distribution of effort to the tenth age group

* + **Expenses**

cand\_num: candidate number

expense: The amount of money the candidate spends

expense\_ceiling: The amount of money the candidate was allowed to spend

* + **PStation**

pd\_num: The unique identifier for the polling station

tot\_votes: Total number of votes for that polling station

cand\_num: The number of the candidate whose votes Pstation.votes will hold

votes: the number of votes the candidate got at this polling station

* + **Link**

pd\_num: the polling station identifier

ctuid: the census tract identifier

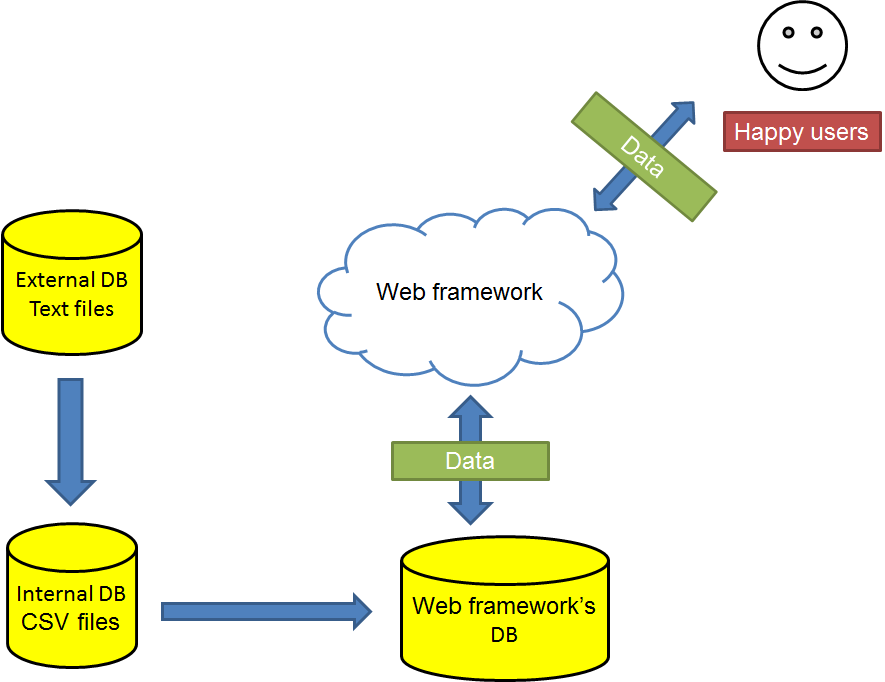
1. **FLOW OF EXTERNAL DATA**

Election data will be collect from external sources. All the external data will be store in

Then all the data will be converted into an internal format in CSV files.

Then all the data will be converted into web framework’s database.

The web framework will use those data for running simulations accordingly to user needs.



*Figure 1: Overview of the flow of data*

1. **URL LISTING:**

**/campaign\_master/**

**GET**: Show the list of campaigns created

**/campaign\_master/add\_campaign/**

**GET**: Show the form to create a new campaign

**POST**: Where to submit data to create a new campaign

/**User/**

**GET:** show the list of campaigns created

**/User/add\_candidate\_effort/**

**GET:** show the form to generate the simulation the outcome of the votes

**POST:** Where to submit data to create a pie chart of the distribution of the votes

**/login/**

**GET:** Show the field for entering the user name and password

**POST:** Sends the user name and password to be verified

**/admin/**

create models and users