

**Team Lion:**

Ana Ashrafi; [ana.a.ashraf@utexas.edu](mailto:ana.a.ashraf@utexas.edu); Github username: anaashrafi  
Austin Duong; [duong\\_austin@utexas.edu](mailto:duong_austin@utexas.edu); Github username: austinduong  
Royce Hong; [rhchen73@utexas.edu](mailto:rhchen73@utexas.edu); Github username: RoyceHong  
Adri Dutta; [adridutta@utexas.edu](mailto:adridutta@utexas.edu); Github username: adridutta  
George Zhang; [zijingzhe@gmail.com](mailto:zijingzhe@gmail.com); Github username: KingGeorgelll  
Phat Le; [phat.le@utexas.edu](mailto:phat.le@utexas.edu); Github username: ple

**Repo URL:** <https://github.com/anaashrafi/Election-Essentials.git>

**Google Doc URL:**

<https://docs.google.com/document/d/1ijS34uwJISdNBwb0nkHS8k3Qe08sMoEblsqx0cXOuJ8/edit?usp=sharing>

**Vision:**

College students don't have time to keep up with the 2020 election and all of the news surrounding it. For most, there's too much going on with classes, extracurriculars, jobs, etc. to be able to do research and learn about each candidate in the race. With our web application, Election Essentials, students would be able to identify the topics that matter most to them, receive information and news articles regarding the various candidates' views on the specified topics. This way, they can be informed and will be more enticed to vote in the upcoming election. In the 2016 election, 43.8% of college students voted (<https://www.insidehighered.com/quicktakes/2017/09/21/study-college-student-voting-rose-2016-election>). The goal of our web application is to increase this percentage. There are of course competitors, such as '20 Matters, Poli Talk, and CommonAlly. However, our app is different because our specific target audience is college students and this will affect the way we want to display our data. Our data would be organized differently than our competitors, which will allow easier and detailed comparisons on different candidates based on the user's essentials. We are also providing not only candidate stance on issues, but also views on these issues from political commentators, gurus from various news outlets, and primary sources. The combinations of these three features cannot be found in any other app, which makes this app competitive.

**Data Sources, Scraping, and Database:**

Data for political topics will be taken from both liberal and conservative websites. For liberal views, websites like Msnbc, Democratic Underground, and Progressive can be used. For conservative views, The Weekly Standard, The federalist, Life Site NewsVoter, and The Blaze can be used. Voter registration information can be taken from usa.gov. For candidate profile page, we can pull data directly from the candidate website, such as donaldjtrump.com, joe Biden.com, betoorourke.com, and kamalaharris.com. For the election news page, the data should be scraped in real time based on the political topic input and source filtrations. Based on the filtered news sources, the political topic will be inputted into the news source url, and only the first article will be displayed to the user for each news source, so as to not overwhelm the

customer with information. For the voting registration information, the data will be scraped once a day because voting registration do not change too much (last update was in august of 2019), and once scraped, it should be stored in our database, so we don't have to depend on the uptime of usa.gov. Along with voting registration laws and regulations, we will be storing voter registration deadlines, so we will know when to notify users who signed up for notifications. Users will be able to sign in using Google and can set their essentials, notification preferences, and view their bookmarked articles. We'll store all of this data so they can log back in and pick up where they left off. Since we don't have many different data types, and have a narrow customer demographic (college students only in USA) we will use MySQL. This is because MySQL is easiest to learn and maintain with small amounts of data, has great uptime, and has minimal delay in updating data.

## **Requirements:**

### **User Description:**

The users who will be using this application are college students who wants information on certain political topics or essentials, and are considering voting in the upcoming presidential election. However, they don't know much about the voting process, nor do they know much about the candidates' viewpoints pertaining to those political topics. In addition, these users are overloaded with information due to extreme amount of news sources.

### **User Stories:**

As an uninformed college student, I would like news articles of different perspectives about a certain political issue or essential, so that I can align myself with a particular perspective.

As a college student with a defined political perspective, I would like information on how certain political candidates approaches certain political topics, so I can vote for them in the upcoming election.

As a college student who is not registered to vote, I would like information on the laws, and procedures of the voting process, so I can vote in the next election.

As a college student, I would like an easy to share information to social media sites, so that I can inform my friends and family members about a certain topic.

As a college student who is particular about my news sources, I would like the ability to filter news sources, so that I feel like I made my choices from sources with integrity.

As a college student who is forgetful, I would like a way to sign up for voting deadlines, so that I can be reminded of when and where I have to vote.

### **Formal Use Case:**

Goal: Get articles from various news sources about a particular political topic

Primary Actor: Uninformed college student

Precondition: On the welcome screen of the webpage

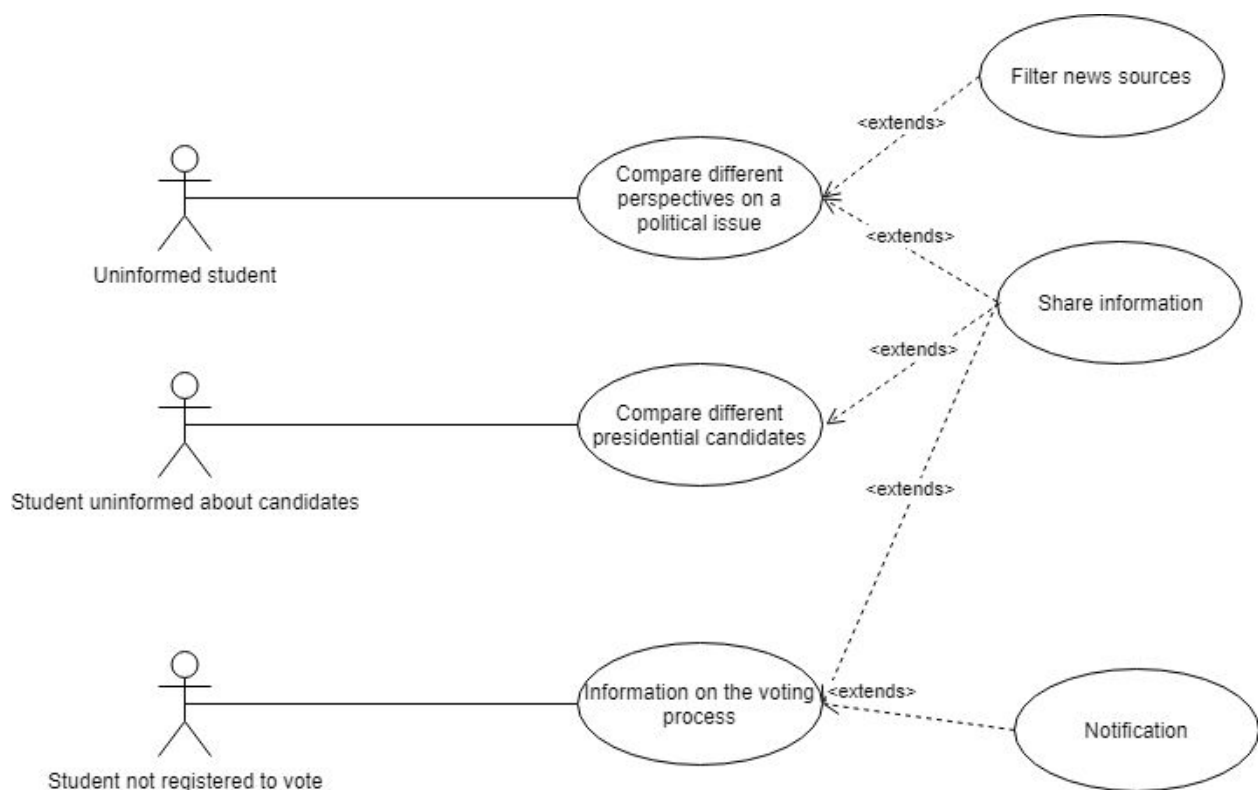
Success end condition: One or more articles about the political topic is displayed and user can share and/or bookmark the article.

Failed end condition: No article about that political topic displayed or user can't share the article.  
 Trigger: Selects the different perspectives tab

1. Given option to login or continue as guest.
2. Select issues from a list of options.
3. Student optionally adds source filtering.
4. System displays articles based on political topic and filtration.
5. Student picks a particular article they find interesting.
6. Student can select social media icon to share article
7. Student enters social media credentials
8. Article is shared on social media platform

Alternatives:

- 3a. No articles can be found based on political topic or filtrations
  - 3a1. System asks student to try different political topics or filtrations
- 6a. Invalid social media credentials
  - 6a1. System notifies student to retry entering credentials



### Interface:

Table 1 on the next page shows four of the main interfaces of the website with color-coded sections. The main page is the first page a user will see when accessing the website. The user is given a list of essentials to check off and submit. The user will then be prompted to create an account or continue as guest. Alternatively, if the user has already registered with the website

they can just log in. The threads will be the general website layout, with navigation in the header. The white section in the threads is where the profiles will be shown. The candidate profiles will show the user relevant articles and social feed along with other candidate information, such as a link to the candidate’s website. The essentials profile will be similar, but without a profile picture. The user profile will contain account settings, voting registration deadlines, and bookmarked articles.

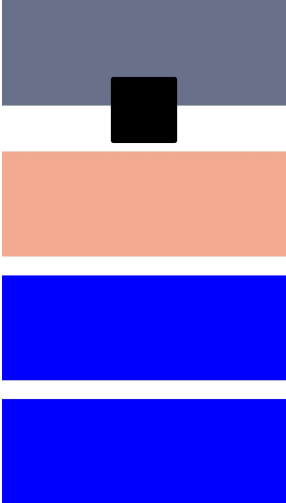
<div>Main Page</div>  <ul style="list-style-type: none"> <li>• Sky blue - header</li> <li>• Light blue - sides</li> <li>• White - main</li> <li>• Brown - footer</li> </ul>	<div>Threads</div>  <ul style="list-style-type: none"> <li>• Sky blue - header</li> <li>• Light blue - sides</li> <li>• White - Profile area</li> <li>• Brown - footer</li> </ul>
<div>Candidate profile</div>  <ul style="list-style-type: none"> <li>• Blue-green - Cover photo/logo</li> <li>• Black - Profile pic of candidate</li> <li>• Tan - More candidate information</li> <li>• Blue - Articles/Social feed</li> </ul>	<div>User profile</div>  <ul style="list-style-type: none"> <li>• Blue-green - Settings</li> <li>• Tan - Voting registration deadline(s)</li> <li>• Light blue - bookmarked articles</li> </ul>

Table 1

**Feasibility:**

Our main concern with implementing Election Essentials is finding reliable sources to data scrape from. Furthermore, we will have to find a way to constantly display recent news in real time so that users can always be informed of what's new. One of the key features of Election Essentials is having data readily available to users in a way that is simple and not overwhelming. With so many candidates, public policies, statistics, and more to display, it will be difficult finding and sorting so many resources to have all of this information at hand for the user. To also push our users to be active and vote, we need to be able to show them places near their location that they can vote at. Since we are scraping the data in real time from other news sources, we are highly dependent on the uptime of those websites. Furthermore, scraping multiple articles in real time might cause high loading time, which can cause our customers to become impatient and leave the website.

**Planning and scheduling:**

Phase	Overview	Details
1	Static Voting Application with ~6 pages hosted on GCP	Learn/Create Rest API calls using Java - 3 hours Create a static website hosted on GCP - 3 hours Obtain a URL from a hostname provider - 1 hour. Learn about HTML and Bootstrap framework - 5 hours Create 6 static pages using HTML and Bootstrap - 5 hours Learn how to scrape data from Github - 5 hours Create the about us section using github data - 5 hours Write Phase I report - 4 hours
2	Data Scraping & Database	Collecting data from various news sources using Python Storing/Organizing/Displaying data Utilize MySQL as primary Database and Javascript to display data on the website. Drafting UMLs Javascript test with Mocha Python unit tests Test the GUI with Selenium Refine report and include UML diagrams.

<b>3</b>	More data and pages, add filtering, searching and sorting to model pages, more testing	Refine our data and website Sort/filter our data based on user's preferences/"essentials" Test web app functionality Application is fully functional after this phase
<b>4</b>	Refactoring & Polishing UI	At least 3 refactorings Trying out 3 different types of designs

#### **Tools, software, frameworks:**

##### **Backend**

Java, MySQL

##### **User Interface**

Javascript, HTML, CSS, Python

##### **IDE & Software**

SQL Server Management Studio, VisualStudio, Eclipse, Git, Mocha, Postman, Selenium

#### **Feedback:**