Develop a Web-Based Election Document Builder

Project Description and Clarification

Votegrity

12-FA25-SP26-VI-WEB
Daniel Calle
Lance Tieng
Ivan Kwok

September 18, 2025

I. Introduction

Votegrity is a platform used for online voting, and they currently use Microsoft Word to build ballots to send out to voters. While it works, it is a slow solution, as every election requires different information in the ballot and may require a different document format. Votegrity is gaining popularity and would like to complete upwards of hundreds of elections per day, and a more streamlined solution is required. To do this, they will need a drag and drop pdf creator that allows internal users to choose from different default formats and have ballots with pre-formatted components.

II. Background and Related Work

Our work is to create a drag and drop pdf creator that allows for streamlined election document creation while also being flexible enough accommodate for any document type. While this is like Google Docs or Microsoft Word, it is more restrictive so that eventually, outside election administrators will be able to create and format their own election documents while conforming to Votegrity's document style. Google Docs and Microsoft Word are both very powerful document creators, and while they are currently used, they are not the ideal solution due to the absolute freedom they provide to the document creator. We will need to learn Django, PostgreSQL, Linux, File Data Structures, React, HTML PDF Converters and Drag & Drop frameworks for this project.

III. Project Overview

The initial problem we are aiming to solve is increasing accessibility and reducing complexity of filling out, creating, and submitting election documents. There exists a variety of election documents within the United States and for some, learning how to properly format or fill out an election document can be a daunting task. By providing a transparent service to fill out or create election documents we can possibly increase voter turnout or possibly allow the running of more elections for whatever is needed. The final product should provide a user-friendly experience that eases the user into creating and editing election documents.

Initial objectives are to study and gain basic knowledge on the necessary technologies and tools necessary for the completion of the project. During this period, we plan on building small features tasked to us by our client. These may be somewhat disjointed, however as we iterate on these features their functionality and cohesiveness should improve immensely after each sprint.

One of our objectives is to handle documents from various user accounts. Our system should be able to convert any file entered a pdf file which can be saved and redistributed across the system. Additionally, we should have proper security measures to ensure users their confidentiality and account security. These security measures will likely be built into user registration. For user management I believe we are using PostgreSQL as our primary database language. Features such as user registration and login should allow us to manage the file creation and saving allowing for efficient creation.

During development we are expected to maintain proper version control throughout all documentation to ensure the accuracy and quality of our documents. Doing so in the early stages of development will ensure that changes are needed and they can be made effectively and efficiently throughout the development process.

Although we should aim for the software to be easily accessible to users, we should have user documentation providing clear guidance on the proper usage and capabilities of the program. Providing these tertiary documents may not assist development, these features will help provide a smoother experience for non-developers and make the distribution and usage of the prototype to create the best possible experience. These guidelines can take many shapes such as a user manual or perhaps possibly provide recommendations on the required fields of entry for each desired document box for the user.

Another goal for us is to develop a front and back end to the website. We plan on using React as our front end and Django as our back-end communicator in a model-view-controller architectural pattern for our development. These tools should allow us to create the framework of the website and allow us to provide our prototype and deliverables. Using this model will simplify the different layers of interaction for our system allowing us to clearly divide the responsibilities of each layer and produce a more synergized system.

We as a group will be learning the multiple roles in development. We each will take turns working on various areas to get a rough understanding of how each part of this application is supposed to work. This not only provides experience in these areas but has us interact with each other's code giving us necessary feedback. Collaborating with other developers will help establish key communication skills necessary for working with a focused group of developers leading to a more cohesive group.

A key requirement for this program is the ability to adapt or interpret the users desired ballot document. The program should allow users to have the freedom to format the document however they like and produce a reusable document for later use. Included in this process we should allow for version control saving previous iterations for easier recovery and history.

IV. Client and Stakeholder Identification and Preferences

Internal Votegrity staff, contractors, people that use internal tools

These are the current stakeholders that create the ballots, and the tool needs to be accessible enough that all created documents follow the same style.

External customers

These are users that would be purchasing the Votegrity service, and they need to be assured that the election document quality is up to par.

Voters themselves

These are people that would be receiving the election documents to vote, and they need to be able to understand what they are voting on and how to from looking at the documents.

Future stakeholder: Non-Votegrity Election Administrator

These are users that would be appointed by customers to administrate the election and create election documents for users. They are not a part of the Votegrity staff and thus may not have the best intentions when creating documents. As a result, they need the tool to be restrictive enough that they cannot make any document they wish.

V. Glossary

Define technical terms used in the document.

PostgreSQL: An open-source relational database management system that uses and extends the SQL language to manage and query data efficiently.

Votegrity: A secure online voting platform aiming to streamline the voting process through digital tools and services.

Django: High-level Python web framework that enables rapid development of secure and maintainable web applications.’

React: JavaScript library for building user interfaces, particularly useful for creating dynamic and responsive web pages.

Model View Controller: A software design pattern that separates application logic into three interconnected components: the model (data), the view (user interface), and the controller (logic that handles input).

Ballot: A document used to cast a vote in an election. In this context, a ballot may be a PDF or digital document customized for different elections.

Election Document: A formal document related to an election process, such as ballots, instructions, or registration forms.

Election Administrator: An individual (either internal or external to Votegrity) responsible for organizing and managing an election, including document creation.

VI.