

Indian Election Database System



Team 54:

Anmol Kumar 2018382

Bhavay Aggarwal 2018384

Diptanshu Mittal 2018232

Manas 2018244

Rishabh Chauhan 2018256

INDEX

- 1) Self-evaluation
- 2) Research and Motivation
- 3) Overview
- 4) Technical Details
- 5) Documentation
- 6) Bonus Part
- 7) ER diagram

Self-Evaluation

- Rishabh Chauhan, 2018256
Through this project, I learned a lot about DBMS. My part in the project was to manage the database and make it easy to access and more efficient. I am satisfied with my work as well as my team's.
- Diptanshu Mittal 2018232
I worked on the backend part of the project which involved fetching queries from the database and updating the database on the client's request. I learned a lot from this project and got to know the advantages of an efficient database. I am happy with my work and my team's work.
- Bhavay Aggarwal 2018384
I worked on managing the database, populating it and making it accessible to the backend developers. I learned a lot from this project .I am satisfied with my work as well as my team's.
- Manas 2018244
I worked on creating and managing the test database and then expanding the main database with data to be utilised by the backend. In all, it was a great experience for me. My team worked really well.
- Anmol Kumar 2018382
In this project I worked on setting up the Bootstrap frontend framework and implementing the backend in Django Framework. I got to learn more by doing this project and it was really worth it. I am satisfied with all my teammates's work.

Research & Motivation

Our initial brainstorming made it clear that there were not many services present which store and present election data to the general public and even news outlets. With this project, we wanted to bridge that gap by providing information like Party Transactions and Advanced Voter Demographics which are neither reported on any news outlet nor are stored by current databases.

Current Databases like <http://www.indiavotes.com> although provide post-election results like voter turnout and close contests, fail to provide any information which might help predict elections. On top of that there is no information about election workers or EVMs. We believe that this data is critical in the smooth functioning of elections and have made sure to include it in our project. We hope that our project will help to create well-informed elections in the times to come and by using our project everyone is able to better analyse their choices.

Overview

We have created an Indian Election Database System which stores the information about the elections, voter, election commission, candidates and political parties. Through our site a user can have access to all the data they need.

Assumption: We have access to all the data about the parties and election.

Our major stakeholders will be Voters, Political Parties, Media Outlets and Independent Researchers.

- For Voters, you will be able to see how much money political parties spent on advertisements using easy to understand graphs. Comparison of different political candidates in education, political history and criminal record. Also, breakdown of party expenditure will help in monitoring the use of tax money and put the misuse of money in the public spotlight.
- For Political Parties, you will get the demographic of voters in all districts on various lines. Also, they can access information about candidates of other parties and can strategize their candidate accordingly.
- For Media Outlets and Independent Researchers, you can combine both the voters and parties data to analyse why a political party was able to win a particular district. Also, by further analysing the data you could give better insight into voting trends. Party expenditure and candidate history data can also be useful for these agencies/individuals.
- For the Election Commission, Records of all the officers involved in the election process will allow for easier administration and will also streamline the allotment process of officers to each district. Also, EVM information will be useful to find and replace defective machines

Technical Design

● Backend

We decided to use Django Framework in our app as it is a python-based web framework and follows the model-template-view architectural pattern which makes it highly scalable.

The main problems that we encountered/anticipated during our project in the backend part were :-

1. How to control the multiple update request to database simultaneously - For this we decided to use locks in our server. Once a thread has acquired the lock, all subsequent attempts to acquire the lock are blocked until it is released. And it was necessary for the threads to acquire the lock before executing the query to the database. Thus the locks ensured that only one client can update the database at a time.
2. Multiple queries at the same time - To overcome this problem we decided to process each client's request on a new thread which helped us to deliver the response faster as the client's request did not require to wait for another request to finish. We noticed a significant decrease in time using the threads for every request.
3. Situation of losing connection to the database - We anticipated that it's quite possible that the connection to the database gets lost and for this made a log table in our server which keeps the entry of each transaction server is doing with the database.
4. Making a cache memory in the server - As the database keeps on getting updates we need to give the clients updated values everytime the value gets updated. So for this we decided to make a cache memory that will get updated whenever someone updates the database. It also helps to save the cost of sending a query to the database for each user as this updated cache is delivered and only one request is sufficient for all the users.

● Frontend

We used web pages , HTML , CSS , Javascript to make the frontend.

The main problems that we encountered/anticipated during our project in the backend part were :-

1. Dynamically updating the graphs and tables on the webpage - We anticipated that the database might be updated by another user while the someone is surfing our website and it won't be possible for the users to keep on refreshing the webpages to get updates. So to overcome this problem we decided to AJAX function to dynamically update the charts and tables for the users bypassing the need to refresh the webpage. Furthermore it's highly efficient as it takes the data from the cache memory from the server and dont require to send the query to the database.

● Database

We have created a Table for each stakeholder to store their details and few tables to establish relation between them. For tables which have large data like voters we have created their index table so that we can access its data quickly. We have also normalized the database so that it is easy to understand and more efficient to update. The database is published in the Microsoft Azure Servers and could be accessed using the Microsoft SQL Server Management Software.

Documentation

The user has to do the following to run this project:

- Since this project is implemented in Django Framework, the user has to install django in his machine and then go to the project directory
`DBMS_PROJECT/django_project`
- After opening this directory, the user has to run the following command:
`Python manage.py runserver`
- After this the user could see the following lines :

```
You have 17 unapplied migration(s). Your project may not work
properly until you apply the migrations for app(s): admin, auth,
contenttypes, sessions.
```

```
Run 'python manage.py migrate' to apply them.
```

```
April 30, 2020 - 14:41:47
```

```
Django version 3.0.5, using settings 'django_project.settings'
```

```
Starting development server at http://127.0.0.1:8000/
```

```
Quit the server with CTRL-BREAK.
```

- Click on the address which is highlighted bold. The project would then open in the browser.

Bonus Part

We have created a virtual election which allows us to estimate the election result before the actual result. A user logs in to our site and chooses the candidate, once he is done he is not allowed to vote again. This way we create a table with the voter details and his vote and after some specific time we declare the estimated results to our users. This way we provide our user the result before the actual votes start counting.

We made a highly efficient server that keeps the track of transaction, maintains a cache memory, keeps the concurrency locks and delivers fast results using multi-threading. Details are mentioned in the technical design.

We made highly efficient web pages which allow the user to keep on getting the updated data without even refreshing the web page. Details are mentioned in the technical design.

ER diagram

Link: <https://app.creately.com/diagram/knHC7u2yS86/>