Setup Description

Raul Persa, Lukas Vogel

January 16, 2016

Used Technology

DBMS: Postgres
We used materialized views for data-aggregation. To refresh them without downtime (by using the `concurrently` keyword), at least version 9.4 is required.
An alternative version of vote-aggregation uses a trigger-based approach. It depends on the new upsert-feature of the postgres 9.5 beta. It is meant as a proof of concept and therefore not enabled by default.

Backend: Django
Django 1.8.6 with Python 3.5 was used during development. Earlier versions should work aswell. Our project also depends on the sslserver and bootstrap3 django packages.

Database-Adapter: psycopg2
We decided against using the Django Object-relational mapper, as it wouldn't give us enough control over our queries. We used the psycopg2 module to talk to the database instead.

Frontend: jQuery and Bootstrap

Load testing: locust
Locust is a Python-module only running on Python 2.

Loading Data Into The Database

Setting up the schema and static data
The database schema is defined in the file `schema.sql`. It can be directly imported into postgres. An easier way to setup the schema is by running the Python3 script `setup.py`. It automatically imports the schema and handles all of the following tasks:

Setting up the vote-insertion functions
The data is imported via `plpgsql`-functions. They are defined in the toplevel-file `voteinsertion.sql`.

Importing views for election algorithm
The views and `plpgsql`-functions needed for the algorithm are defined in the file `election-algorithm.sql`.

Importing views for analysis of Wahlkreise
Some assorted views to facilitate queries for the Wahlkreis-Overview web page are defined in the file `wahlkreis-analysis.sql`.

Importing static data Data like parties, Bundesländer, Wahlkreise, Candidates, etc. is extracted from the files in the `data/` directory and imported by `setup.py`.

Generate votes
The Python3 script `votegenerator.py` generates the right amount of voters and votes for the elections of 2009 and 2013 and imports them into the database. It takes advance of the `plpgsql`-functions defined in `voteinsertion.sql` already imported by `setup.py`.

Running the Wahlsystem

The subdirectory `datenbanken-app` is a django-project. It consists of the two applications `wahlanalyse` handling the analysis part of the system and `wahl` handling the actual voting.

It can be started by executing:
`python datenbanken-app/manage.py runserver localhost:8000`

The Wahlinformationssystem should then be reachable at:
`localhost:8000/wahlanalyse/2/` or just: `localhost:8000`
where 2 is the election id.