Парсер-выгрузка данных по выборам в России из ЦИК

Репозиторий проекта:

https://github.com/EmilStasevski/Parsing-CEC-Central-Election-Commission-of-the-Russian-Federation

Отчет о работе

Что сделали между КТ-1 и КТ-2 *тезисно*:

- поменяли архитектуру с flask + dash на selenium/bs4 + Firebase (в качестве БД) + React на JS (с разрешения лектора)
- привели репозиторий в порядок (добавили документацию, разделили папки с кодом, добавили requirements.txt, .gitignore, описание каждого из разделов, примеры данных (выгрузок), добавили драйвер chromedriver, чтобы при клонировании репозитория можно было провалидировать результаты без поиска дополнительных зависимостей)
- проанализировали, как устроен сайт ЦИКа с результатами выборов (http://www.vybory.izbirkom.ru/), продумали "пользовательский путь": какие фильтры в каком порядке обычно ставят, как имитировать поведение пользователя, чтобы нас не забанили / не дали ввести капчу
- написали backend-часть, используя selenium (удаленное открытие Google Chrome с имитацией под человека, кликание по сайту "как человек" для выбора нужных результатов выборов), bs4 и lxml (парсинг нужных страниц, забирание требуемых данных)
- написали драфтовую версию сайта нашего приложения: https://zj523c.csb.app/
    - в драфтовой версии сайта есть 4 страницы (по 4 типам выборов), к 2 из них подгружены первые данные (для президентских выборов данные за 2012 и 2018 год, для голосований по поправкам в Конституцию в 2020 г. данные по Республике Адыгея), и для них же есть скелет первых графиков
    - backend и frontend соединены "искусственно" — они читаются из гитхаба, деплоятся json-формате в Firebase. Внутри нашей базы данных предполагается наличие 4 коллекций датафреймов (пока есть по президентским выборам и по поправкам в Конституцию). Пример можно увидеть здесь
    - выбраны базовые цвета дизайна сайта в файле style.css

Что осталось:

- масштабировать парсер (backend-часть) на все возможные сценарии выбора фильтров и уровни детализации информации
    - пока что в целях тестирования и отладки работали с ограниченным пулом сценариев
- доделать визуал frontend-части
    - пока что больше уделяли внимание функционалу (в частности, нужно добавить бар чарты и карты на всех 4 страницах приложения, а также обновить главную страницу)
- “поженить” backend и frontend части, чтобы сервис работал end-2-end.
    - на данный момент есть “костыль” с передачей данных от парсера в приложение на React

Кто что делал:

- *Сташевски Эмиль*: поднятие и настройка Firebase, вся frontend-часть (принятие данных от парсера, обработка и отрисовка, визуальные фильтры на странице приложения для пользователей), взаимодействие backend- и frontend-частей, дизайн приложения (css)
- *Рубанов Владислав*: организация репозитория, документация, анализ сайта ЦИК (хранение данных в html-коде после выбора фильтров, парсинг таблиц с результатами выборов), backend-часть (работа с selenium (поиск нужных результатов после первичного выбора фильтров), написание парсера, скриптов и тестирование, взаимодействие backend- и frontend-частей)
- *Жужлев Борис*: анализ сайта ЦИК (от стартовой страницы до выбора первичных фильтров), backend-часть (работа с selenium (выбор первичных фильтров))