

Тестовое задание (python)

Реализовать веб-приложение для поиска по Stack Exchange ([api](#)).

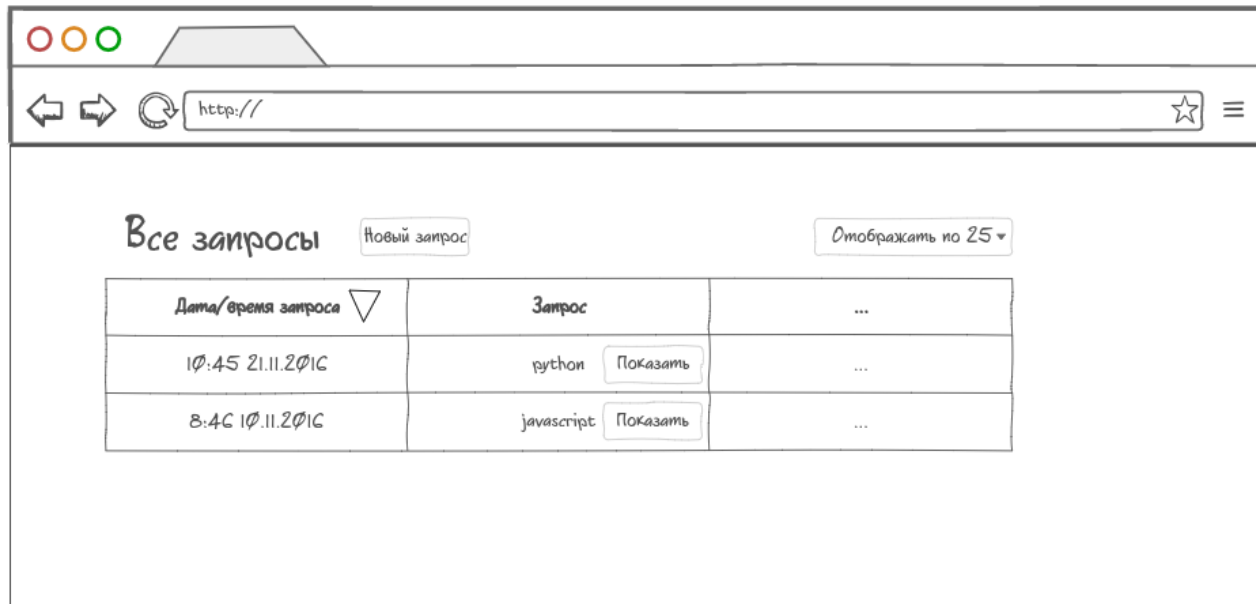
Условия:

- Рекомендуется самостоятельно реализовать запросы к API, [без привлечения готовых библиотеки с рурі](#).
- Использовать framework `aiohttp/pyramid`.
- Рендеринг серверный, никаких `ajax` запросов от `javascript` библиотек.
- Результаты запросов записывать в реляционную БД (`mysql`, `postgresql`, можно `sqlite`). Структуру БД разработать самостоятельно, разбить на таблицы, определить основные ключи.
- Результаты запросов выводить постранично, на странице можно указать 25, 50 и 100 (при изменении опции количества строк делать перезапрос на ответы к БД или напрямую к API stack exchange).
- Сохранять не обязательно все результаты сразу, дописывать можно при переходе на "следующую" страницу ответов.
- Результаты можно сортировать (сортировка только по одной колонке, при изменении сортировки делать перезапрос на ответы к БД или напрямую к API stack exchange).
- Реализовать 1 минутное кэширование `http` ответов на запросы к приложению через `redis`.
- Реализовать фоновый обработчик, который раз в 10 минут обновляет информацию в БД по типам запросов, которые уже сохранены (выполнялись ранее).
- Через `websocket` отправлять в браузер уведомления о том, по каким запросам информация была обновлена (со стороны `javascript` использовать нативный полифилл [WebSocket](#)).
- Использовать принципы `pip/setuptools` для установки зависимостей (никаких системных зависимостей, только установка пакетов в отдельный `virtualenv`).
- Описать как нужно запускать фоновый обработчик.

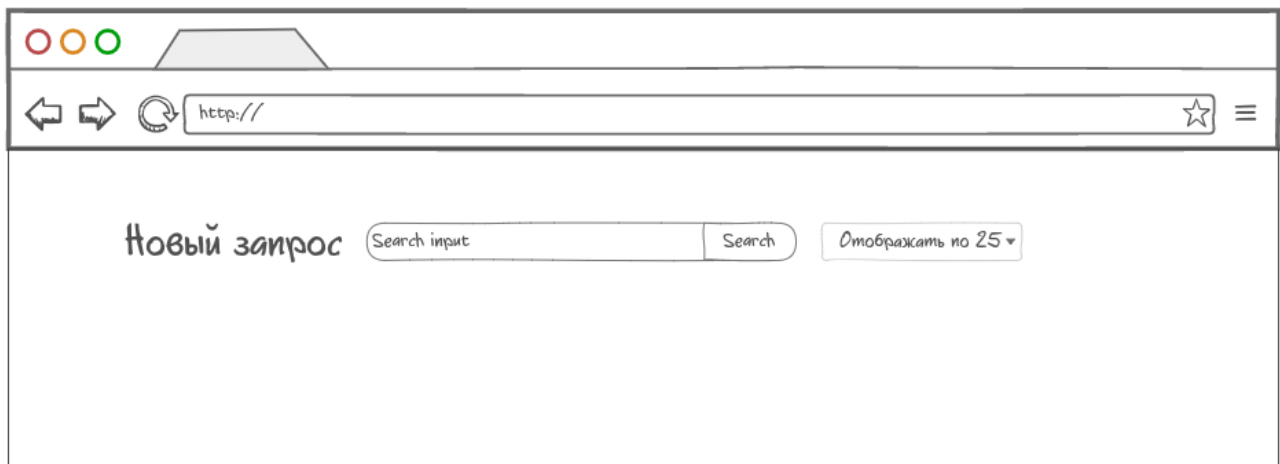
Все конкретные значения, которые описаны выше рекомендуется вынести в `ini` файл настроек (таймаут кэширования, таймаут фонового обработчика, постраничность, сортировка)

Ориентировочный внешний вид

Интерфейс списка всех запросов:



Интерфейс начала поиска (новый запрос):



Интерфейс отображения запросов по теме:

