

# Python разработчик

Дэшборд моделирования функций.

## Задание

Необходимо разработать приложение, которое позволит пользователю моделировать работу функций  $y=f(t)$  и анализировать соответствующие графики.

Пример функции:  $y=t*t + 2/t$ , где

$t$  - unixtime в интервале  $[\text{datetime.now()} - \text{interval}, \text{datetime.now()}]$  с шагом  $dt$ ,

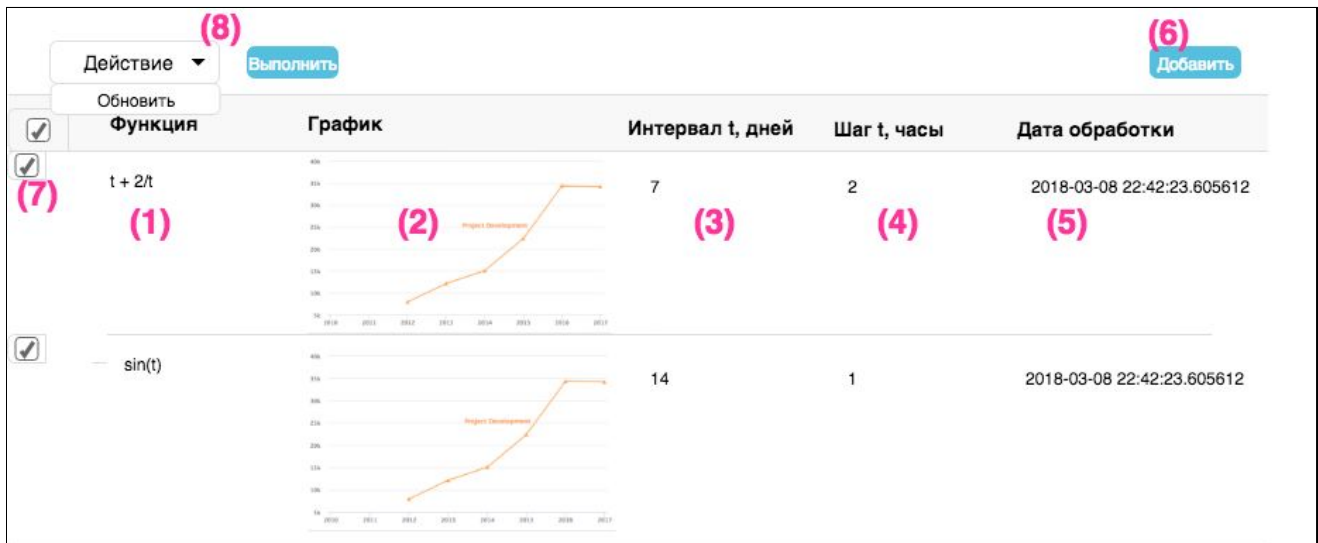
**Interval** - глубина периода моделирования в днях

**datetime.now()** - дата обработки

**dt** - шаг в часах

## User story

1. Пользователь заходит в админку, видит список ранее сохраненных функций (1), соответствующие графики (2), дату последней обработки (5), параметры (3) и (4), а также кнопку добавить (6).



Действие	Функция	График	Интервал t, дней	Шаг t, часы	Дата обработки
Обновить	(7) $t + 2/t$ (1)	(2)	(3) 7	(4) 2	(5) 2018-03-08 22:42:23.605612
	$\sin(t)$		14	1	2018-03-08 22:42:23.605612

\* графики на скриншоте представлены для примера и не отражают представленные функции

2. Пользователь жмет кнопку добавить (6), выбирает одну из двух предустановленных функций (1):  $t + 2/t$ ;  $\sin(t)$ , а также значение  $dt$  и  $interval$ . Введенная информация сохраняется в БД.
3. Приложение генерирует изображение для введенной функции и сохраняет результат в БД. Если возникает ошибка - то сохраняется текст исключения (он выводится вместо графика). После обработки обновляется поле дата обработки (5).
4. По завершению обработки вновь открывается список (1), в которой появляется строка с только что введенной функцией, пользователь видит в строке с функцией график или текст исключения, введенные параметры и дату обновления.

5. Пользователь выбирает галкой строки с ранее введенными функциями (7) и запускает действие “Обновить” (8).
6. В фоне запускается параллельное обновление изображений для выбранных строк.
7. Пользователь самостоятельно обновляет страницу в браузере и через некоторое время видит обновленные графики с обновленным полем “Дата обработки”.

## System story

Приложение состоит из трех сервисов:

1. Сервис1 - админка, отвечает за взаимодействие с пользователем, хранение и отображение пользователю информации о введенных функциях и сгенерированных графиках.
2. Сервис2 - генератор данных, отвечает за генерацию данных для графика. Принимает одну из двух предустановленных функции, генерирует по ним массив [(x, y)].
3. Сервис3 - генератор изображений, генерирует изображение по подготовленным данным.

Сценарий взаимодействия при добавлении новой функции:

1. Сервис1 сохраняет инфу о функции в БД и запускает операцию получения изображения.
2. Операция получения изображения сервиса1 обращается в сервис2 через REST API, передавая функцию в текстовом виде и параметры.
3. Сервис2 обрабатывает запрос, генерирует точки для графика по переданным данным.
4. Сервис1 получает точки и передает их в сервис3 по REST API.
5. Сервис3 принимает набор точек и возвращает изображение.
6. Сервис1 получает изображение и сохраняет в БД, обновляет дату обновления.

Сценарий параллельного обновления через действие в админке нужно спроектировать самостоятельно.

## Требования к реализации

1. Все сервисы приложения разместить в одном репозитории github или bitbucket. Название репозитория должно состоять из рандомного хэша.
2. Для организации окружения и взаимодействия сервисов использовать docker-compose.
3. Для реализации прикладных сервисов использовать python.
4. Для АПИ и админки допустимо использовать django, drf, flask.
5. Генерация точек для графика должна осуществляться в SQL запросе. Те массив точек должен быть получен путем вызова SQL стейтмента. Возможны различные варианты решений для удовлетворения данного требования.
6. Для сервиса генерации изображений использовать возможности готового образа [highcharts](https://hub.docker.com/r/highcharts/highcharts).
7. Для БД использовать Postgresql.

8. Для организации фоновых задач использовать celery с шиной сообщений RabbitMQ и бэкендом результатов Redis.
9. В корне репозитория создать README.md, с описанием system stories и информацией о том как запустить окружение для их проверки.

## Работа над заданием

- Перед стартом работ сделать оценку по времени и срокам выполнения и отправить оценку письмом на адреса
- В теме письма необходимо указать название вакансии, а в письме свои данные и оценку по срокам выполнения задания
- прислать ссылку на электронную почту на адреса ,

## Если возникают вопросы

- По всем вопросам пишите в цепочку с подтверждением задания ,

С нетерпением ждем джедая микросервисов и строителя дата пайплайнов)