Тестовое задание на вакансию "техлида курса Python-разработчик" в Skypro

Сделайте копию этого файла и выполните задание в ней. Пожалуйста, не стирайте задания — делайте прямо под ними.

Фамилия, имя	Шаульский Александр
Кем сейчас работаете? Если не работаете, то укажите опыт работы и причину ухода с последнего места.	5+ лет в EdTech в школе Helloworld (бэкенд разработчик и немного преподаватель).
Укажите ссылки на 2–3 примера работ, которыми вы гордитесь, и поясните, почему именно ими.	По понятным причинам я не могу показывать коммерческие проекты. Мой код можно посмотреть вот тут вот: https://github.com/aleksandrshaulskyi?tab=repositories
Расскажите, что для вас важнее всего в работе, а с чем вы не будете мириться ни в коем случае.	Не буду мириться с аморальным и незаконным, причем приоритет в этом порядке. Остальное - окей.
Есть ли у вас опыт работы в EdTech? Если да, то какой? С какими форматами и где работали?	Да. Онлайн уроки 1/1, но, в основном, конечно, именно разработка.
Был ли у вас опыт руководства авторами? Если да, то какой?	Авторами - нет. В целом - да. Есть опыт авторства в продажах. Несмотря на то, что я уже 7 лет в них не работаю, мои тексты до сих пор используют и они, очевидно, приносят деньги.
Email (телеграм)	@ashowlsky

Что нужно знать перед тем, как сесть за тестовое: *мы общаемся на «вы», но без канцеляризмов, штампов и делового стиля. Работа будет в паре с редактором и дизайнером, поэтому мы не будем обращать внимание на ошибки, запятые и все картинки и схемы для тестового можно взять из интернета или отрисовать черновой вариант самому.*

Нам важно посмотреть насколько интересным, структурным и доступным будет урок.

Задание 1. Написать урок “с нуля”

1. Выберите тему урока по вашей специальности которая вам больше всего нравится. Желательно чтобы урок смог понять человек не сильно погруженный в профессию. Не нужно писать весь урок. Сделайте часть.
2. Напишите план и текст для урока. Приложите схемы, картинки. В итоге у вас должен получится черновой конспект, по которому будут работать редактор и дизайнер.

Ссылка на мой урок: Lesson 1

Задание 2. Техническое задание

Задача

Необходимо реализовать проект, который ответственный за проверку исходного кода проекта, причем в сервисе необходимо реализовать API прослойку для автоматизации работы компаний партнеров. Результат прислать на почту указанную в шапке документа.

Основные модули, из которых состоит система

1. Модуль авторизации и регистрации пользователя.
Пользователь должен иметь возможность зарегистрироваться в системе по паре почта и пароль, а также войти по этим данным.

2. Модуль загрузки файлов с исходным кодом.
Авторизованный пользователь должен иметь возможность загрузить файл в систему, при этом информация о загруженном файле должна сохраниться в базу данных с пометкой о том, что файл новый. Помимо этого, пользователь должен иметь возможность удалить файл или перезаписать, таким образом, в базе данных должны быть соответствующие пометки. Очень важно не давать загружать файлы, у которых расширение не равно “.py”.

3. Модуль проверки соответствия кода общепринятым правилам.

По расписанию выполняется задача на автоматическую проверку кода для новых загруженных или перезагруженных файлов. Проверку рекомендуется проводить с помощью утилиты flake8, чтобы обнаружить несоответствия общепринятым правилам оформления кода на python. По итогу проверки необходимо сформировать отложенную задачу на отправку письма пользователю с информацией о проведенной проверке. Важно хранить лог каждой проверки для каждого файла, который находится в списке файлов у пользователя в системе.

4. Модуль отправки письма с уведомлением пользователю.

Необходимо реализовать обработку задач из очереди на отправку уведомлений пользователю о результате проверки его файла. При этом, важно в логах проверки отмечать факт отправки сообщения пользователю.

5. Модуль отчета о проведенных проверках

В системе также должен быть интерфейс, в котором пользователь может просмотреть результаты выполненных проверок с пометками об отправке отчета пользователю на почту.

Рекомендации

- Реализовать интерфейс можно с помощью uikit Bootstrap
- Для работы с очередями лучше использовать celery в связке с redis
- Для запуска периодических задач можно использовать celery beat

API

Необходимо реализовать эндпоинты для управления проверками или для перезапуска этих проверок. Всего должно добавиться 3 новых запроса, которые реализуются на основе пакета DRF: список файлов пользователя, информация по одному файлу, запрос на повторную проверку файла.

Список файлов

Адрес запроса:

GET: /api/files/

Пример ответа:

```
[
	{
		"id": 1,
		"filename": "myfile.py",
		"last_check": "2023-02-01",
		"status": "success"
	}
	...
]
```

Информация по файлу

Адрес запроса:

```
GET: /api/files/1/
```

Пример ответа:

```
{
	"id": 1,
	"filename": "myfile.py",
	"last_check": "2023-02-01",
	"status": "success",
	"checks": [
		{
			"status": "done",
			"date": "2023-01-02",
			"result": "..."
		}
		...
	]
}
```

Повторная проверка

Адрес запроса:

```
POST: /api/files/1/
```

Пример ответа:

```
# 200
{
        "result": "ok",
        "message": "file under testing"
}

# 403
{
        "result": "failed",
        "message": "have no access to this file"
}
```

Примечания

1. Набор полей и их названия в вашем проекте и в примере могут отличаться, в примере передается суть того, какие данные должны передаваться.
2. В реализации использовать как Generic классы, так и ModelViewSet

Критерии решения

- Интерфейс системы должен содержать следующие экраны: вход, регистрация, список загруженных файлов, отчет проверки по каждому файлу отдельно, изменение файла, удаление файла
- Реализованы все 5 модулей
- Для разных сервисов созданы отдельные контейнеры (django, postgresql, redis, celery, при необходимости список можно самостоятельно расширять), все оформлено в docker-compose файле, при необходимости можно создавать вспомогательные Dockerfile
- Проект готов быть размещен на удаленном сервере
- Интерфейс понятен и соответствует базовым требованиям системы
- Решение выложено на github.com

Дополнительные задания

1. Напишите тесты для основных функциональностей

2. Сделайте верификацию после регистрации через почту (отправьте письмо со ссылкой пользователю на почту, чтобы он подтвердил таким образом свою почту)

3. Перечислите другие варианты создания периодических задач и реализуйте как минимум один дополнительный

4. Вынести все пароли и доступы в переменные окружения, записать переменные окружения в образ через docker-compose

Ссылка на гит: https://github.com/aleksandrshaulskyi/test-file-checker
В ридми вся информация.