Тестовое задание Python

# Описание:

Необходимо реализовать и уметь локально развернуть web server, отдающий html страницу при переходе на домашний адрес сервера в браузере а также принимать три типа запросов и возвращать ответы в формате JSON:

Требования к web-серверу:

- Сервер должен быть реализован с использованием одного из популярных web-фреймворков языка Python (Flask, Django, Tornado, Sanic и т.п.).

- Сервер должен уметь обрабатывать запрос GET /orders и возвращать JSON вида:

[

{

“id”: 1,

“user\_id”: 1,

“order\_info”: “Заказ 1”

},

{

“id”: 2,

“user\_id”: 1,

“order\_info”: “Заказ 2”

}

…

]

Со всеми имеющимися сущностями «order».

- Сервер должен уметь обрабатывать запрос POST /order c телом запроса, содержащим JSON вида:

{

‘’user\_id”: 1,

“order\_info”: “Заказ N”

}

И возвращать JSON ответ:

{

“status”: 1

}

Если валидация пройдена успешно. Иначе:

{

“status”: 0

}

При обработке запроса, необходимо удостовериться, что тело запроса содержит поля «user\_id» и «order\_info», а также, что значение поля “user\_id” равно единице.

При успешном выполнении запроса полученные данные должны сохраняться.

- Сервер должен уметь обрабатывать запрос GET /order?order\_id=N, где N – идентификатор заказа (id) и возвращать ответ вида:

{

“id”: N,

“user\_id”: 1,

“order\_info”: “Заказ N”

}

Если такой “id” есть, или возвращать ответ со кодом состояния HTTP равным 400 (Bad Request - «неправильный, некорректный запрос»).

Где возвращаемая сущность “order” соответствует сохраненным данным о ней.

# Требование к хранению данных:

- Хранение данных может быть реализовано на усмотрение разработчика, это может быть, как и База данных так и локальный json или txt файл.

# Справка:

- Для удобства тестирования запросов можно воспользоваться утилитой Postman или url строкой браузера.

- Требований к содержимому отдаваемой html страницы нет.