

Программа веб-сервиса, способного запускать и останавливать работа

Тестовое задание на позицию Python-разработчик RPA (стажер).

Разработала:

Новохатская Александра Игоревна

Задание выдано: 25.03.2024

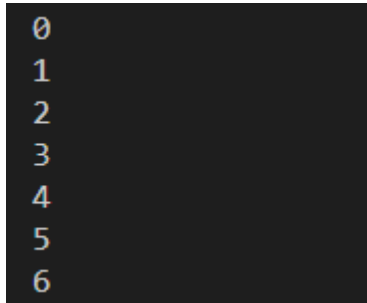
Дата завершения: 28.03.2024

Москва, 2024

Задание:

Необходимо написать бекэнд веб-сервиса на FastAPI, который способен запускать и останавливать робота.

1. Робот – это Python-скрипт, который также необходимо написать. Он должен работать отдельно и независимо. Его задача: каждую секунду выводить в консоль цифры от 0 и далее, прибавляя по единице, пока выполнение скрипта не будет прервано. Робот должен работать асинхронно.



```
0
1
2
3
4
5
6
```

Робот при запуске должен принимать параметр командной строки – с какого числа начинать отсчет (по умолчанию – с нуля). В веб-сервисе на FastAPI должна быть возможность передать это число.

2. Программа должна хранить информацию о времени и длительности каждого запуска в базе данных (SQLite), а также информацию о том, с какого числа был начат отсчёт. FastAPI должен иметь дополнительный эндпоинт для вывода этой информации.

Дополнительные требования:

1. Программа должна работать и запускаться под Windows.
2. Перед началом работы необходимо инициализировать git-репозиторий, чтобы мы могли посмотреть, как шла работа над проектом.

Описание элементов программы:

robot.py

Программа робота каждую секунду выводит в консоль цифры от *n* и далее, где число *n* передается через командную строку. По умолчанию *n* равняется нулю.

main.py

Бэкэнд веб—сервиса на FastAPI, способный запускать и останавливать робота, а также выводить информацию о времени запуска, продолжительности и начального числа *n* для каждого запуска.

robot_history.db

Создаваемый программой файл, содержащий таблицу с данными о времени запуска робота, продолжительности его работы и числе, с которого программа начинала отсчет.

create.sql, insert.sql, show.sql

Файлы, содержащие SQL-запросы, необходимые для работы программы.

Инструкция к программе main.py:

1. Основная страница.

При запуске программы *main.py* появится ссылка на сервер: *http://localhost:8000*.

При успешном переходе на веб-сервер пользователь увидит следующее сообщение:

```
{"message": "You can start_robot, stop_robot or see_history"}
```

2. Запуск робота.

Для запуска робота (программы *robot.py*) через веб-сервис необходимо перейти на страницу *http://localhost:8000/start_robot*.

При успешном запуске пользователь увидит сообщение:

```
{"message": "Robot started from number {start_number}"}
```

Если робот уже был запущен, пользователь увидит сообщение:

```
{"message": "Robot has already started from number {start_number}"}
```

Для того, чтобы робот начинал отсчет с определенного числа *n*, отличного от нуля, необходимо передать число как параметр в следующем формате: */start_robot?start_number=n*.

3. Остановка работа.

Для остановки работа необходимо перейти к *http://localhost:8000/stop_robot*.

При успешном выполнении пользователь получит сообщение:

```
{"message": "Robot was stopped"}.
```

Если робот не был запущен, пользователь увидит сообщение:

```
{"message": "Robot wasn't working"}.
```

4. Просмотр данных о запусках работа

Для просмотра базы данных, содержащей историю запусков (начальное число, время запуска, продолжительность работы программы в секундах), необходимо перейти к *http://localhost:8000/see_history*.

При успешном выполнении пользователь увидит содержимое базы данных в следующем формате:

```
{"history": "[('5', '2024-03-28 08:06:18.867132', '11')]"}
```

, где элементы — начальное число, дата и время запуска и продолжительность работы робота соответственно.

Если таблица не была создана, пользователь увидит сообщение:

```
return {"message": f"There is no history yet"}.
```

Ход работы:

1. Для начала была разработана программа робота (robot.py).

Используемые модули:

asycio — для создания асинхронных функций.

sys — для взаимодействия с параметрами командной строки.

Функции:

main() — обработка данных, переданных через командную строку, и запуск функции robot.

robot(start_number) — асинхронная функция, содержащая бесконечный цикл, считающий от n до остановки.

2. В файле main.py был написан код, запускающий веб-сервер.

Используемые модули:

`fastapi` — веб-фреймворк для создания API.

`uvicorn` — библиотека для запуска веб-сервера.

Функции:

`root()` — вывод сообщения на первую страницу.

3. Затем были разработаны функции для запуска и остановки робота.

Используемые модули:

`subprocess` — соединение с потоками стандартного ввода и вывода для запуска и остановки программ.

Функции:

`start_robot(start_number)` — запуск робота.

`stop_robot()` — остановка робота.

`has_started()` — проверка того, запущен ли робот в данный момент.

4. Последним был разработан код для создания, заполнения и вывода базы данных.

Используемые модули:

`sqlite3` — библиотека для работы с SQLite.

`datetime` — модуль для работы с датой и временем.

Функции:

`work_with_db()` — создание и заполнение базы данных.

`see_history()` — создание эндпоинта FastAPI для вывода содержимого базы данных.

Потенциальные изменения:

В ходе работы было принято решение не использовать `raise HTTPException()`, так как выбор недоступной команды не мешал работе программы. В случае усложнения функционала веб-сервиса может быть необходимо ввести данную меру.

Заключение:

Был разработан бекэнд веб-сервиса на FastAPI, способный запускать и останавливать робота, а также показывать данные об истории его запусков.

Код программы был размещен на платформе GitHub:
<https://github.com/AlexNovokh/GreenAtomTask>.