Приложение работает следующим образом:

**Эндпоинт** /generate\_token:

* 1. Генерирует случайный API токен и возвращает его пользователю.
  2. Токен сохраняется в списке api\_tokens в хэшированном виде.

**Эндпоинт** /analyze:

* 1. Ожидает POST-запрос с JSON-данными, содержащими URL сайта и фразу для поиска.
  2. Проверяет наличие и валидность API токена в заголовках запроса.
  3. Выполняет анализ сайта на наличие фразы и возвращает результат в виде JSON.
  4. Логирует запрос и результат.

### Как использовать приложение:

**Запустите приложение**:

python microservice.py

Приложение будет работать по умолчанию на http://127.0.0.1:5000.

**Получите API токен**:

curl -X POST http://127.0.0.1:5000/generate\_token

Ответ будет содержать API токен в JSON формате:

{

"token": "your\_generated\_token"

}

**Используйте API токен для анализа сайта**:

curl -X POST http://127.0.0.1:5000/analyze -H "Content-Type: application/json" -H "Authorization: your\_generated\_token" -d '{"url": "http://example.com", "phrase": "искомая фраза"}'

Ответ будет содержать результат анализа:

{

"phrase\_found": true,

"levenshtein\_distance": 0

}

### Логирование и сохранение ответов:

**Логи**:

* 1. Логи по умолчанию выводятся в консоль и содержат информацию о параметрах запроса, результате и времени обработки.
  2. Вы можете перенаправить логи в файл, изменив конфигурацию логирования:

logging.basicConfig(level=logging.INFO, format='%(asctime)s %(levelname)s %(message)s', filename='microservice.log', filemode='a')

**Кэширование**:

* 1. Кэширование результатов анализа осуществляется с помощью cachetools.TTLCache.
  2. Результаты кэшируются в памяти приложения и автоматически удаляются по истечении времени жизни кэша (60 минут/3600 секунд).

### Пример изменения конфигурации логирования:

Чтобы перенаправить логи в файл, измените конфигурацию логирования в начале скрипта:

import logging

logging.basicConfig(level=logging.INFO,

format='%(asctime)s %(levelname)s %(message)s',

filename='microservice.log',

filemode='a')

Теперь все логи будут сохраняться в файл microservice.log в директории, где запущено приложение.

### Пример анализа запроса и логирования:

def log\_request(params, result):

logging.info(f"Params: {params} - Result: {result}")

@app.route('/analyze', methods=['POST'])def analyze\_endpoint():

if 'Authorization' not in request.headers:

abort(401)

token = request.headers['Authorization']

hashed\_token = hash\_token(token)

if hashed\_token not in api\_tokens:

abort(403)

data = request.get\_json()

if not data or 'url' not in data or 'phrase' not in data:

abort(400)

url = data['url']

phrase = data['phrase']

result = analyze\_website(url, phrase)

log\_request({"url": url, "phrase": phrase}, result)

return jsonify(result)

Теперь запросы и результаты будут логироваться и сохраняться в указанном файле логов.