**Несложный API для обработки графа на Node.js**

**Как я подружил API на Node.js и Python для обработки графа.**

Эта статья об разработке архитектуры API на Node.js, используя веб-фреймворк Express и обработки графа на языках JavaScript и Python.

Глоссарий

API (Application Programming Interface) — это набор определенных методов и инструментов, предоставляемых программным обеспечением для взаимодействия с другими программами. Разработка API является важной частью процесса создания веб-приложений и сервисов, и Node.js является одной из популярных платформ, используемых для разработки API.

Node.js — это среда выполнения JavaScript, построенная на базе движка V8 JavaScript Engine от Google. Благодаря своей асинхронности и событийно-ориентированной архитектуре, Node.js обеспечивает высокую производительность и масштабируемость, что делает его идеальным выбором для разработки API.

npm (Node Package Manager) — это стандартный менеджер пакетов, автоматически устанавливающийся вместе с Node.js. Он используется для скачивания пакетов из облачного сервера npm, либо для загрузки пакетов на эти сервера.

Express - это минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для мобильных и веб-приложений.

Endpoint - Это – название адреса, который будет принимать сообщения от клиента. Обычно представлен URL и портом. Пример: http:/127.0.0.1:5000

Перед началом разработки API, необходимо задаться вопросом: Какую задачу оно будет решать? Конечно же обработку каких-то данных. Для примера я выбрал обработку не ориентированного графа, а точнее получение соседей выбранного узла.

Почему я выбрал Node.js для разработки API?

Существует несколько преимуществ использования Node.js для разработки API:

1. \*\*Высокая производительность\*\*: Node.js обеспечивает низкий уровень нагрузки на процессор благодаря асинхронной обработке запросов. Это позволяет обрабатывать большое количество параллельных запросов без затормаживания системы.

2. \*\*Единая языковая среда. Они также готовы помочь в решении проблем и дают возможность быстро найти ответы на вопросы.

Начало разработки на Node.js

Прежде чем приступить к разработке API на Node.js, убедитесь, что у вас установлен Node.js на вашей системе. Вы можете скачать и установить последнюю версию Node.js с официального сайта (https://nodejs.org).

Как только Node.js установлен, вы можете создать новую директорию для вашего проекта и инициализировать его с помощью npm. Откройте командную строку (или терминал) и выполните следующие команды:

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Автоматически созданное описание

При выполнении команды `npm init` вам будет предложено задать несколько вопросов о вашем проекте. Вы можете просто нажать Enter, чтобы принять значения по умолчанию, или ввести свои значения.

Сразу установим необходимые пакеты для проекта:

1. express – пакет фреймворка
2. express-fileupload – дополнительный пакет для загрузки файлов в Express
3. nodemon – утилита которая отслеживает файловую систему приложения Node и автоматически перезапускает процесс, если есть такая необходимость

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Далее настраиваем запуск сервера Node в файле package.json

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Далее создаем точку входа в приложение

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Запускаем приложение командой:

Изображение выглядит как снимок экрана, Шрифт, текст, Графика

Автоматически созданное описание

Заходим по адресу http:/127.0.0.1:5000 и видим всеми нами любимое сообщение:

Изображение выглядит как текст, снимок экрана, Шрифт, белый

Автоматически созданное описание

Декомпозиция API.

Для того чтобы сохранить читаемость кода нашего API и не плодить много функций в одном месте подготовим папки и файлы для его декомпозиции.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

1. controllers – директория для контроллера ненаправленного графа
2. module\_gm – директория для классов графа на языке js и python
3. routes – директория для endpoint`ов
4. storage – директория для сохранения загруженных файлов и результатов выполнения API
5. test\_files – директория для файлов с тестовыми данными графа
6. utils – директория для переиспользуемых функций.

Вспомогательные функции для работы с csv и json

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Класс UnDirectedGraph

Реализуем простой класс ненаправленного графа с базовыми функциями получения получения узла и ребра, а также функцию с алгоритмом нахождения соседей выбранного узла на языке js

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Контроллер и роутинг API



Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

После создания контроллера и роутов модифицируем index.js файл

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Подготовка тестовых данных

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Тестирование. Postman.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, линия, Шрифт, белый

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, диаграмма, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание

Добавления python

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Добавляем функцию запуска python скрипта из js в utils.js

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, часы

Автоматически созданное описание