Sudoku W/ Ease
Lab 1 Report

Yichen Guo yg2258@nyu.edu

06/14/23

Experiment 1: We will be using Javascript's execution time command to see the time difference between using RESTful API and GraphQL. Clear the Sudoku after each solution and generate a new solution for the blank Sudoku. Draw a table as follows and for each entry calculate the execution time (in ms). Calculate the average execution time.

Sudoku size	Trial #	RESTful API	GraphQL
4×4	1	20.5791015625	20.6689453125
	2	22.39794921875	24.559814453125
	3	23.035888671875	23.287841796875
	4	21.14892578125	23.45703125
	Average	21.790	22.993
9×9	1	34.7109375	40.56494140625
	2	32.298095703125	39.533935546875
	3	35.0849609375	41.416259765625
	4	34.859130859375	39.635009765625
	Average	34.238	40.288

• Which one is faster? Why?
Answer:
In this case, RESTful API is faster.

In this case, because we are making requests to local Flask backend APIs, and each request will call a local Flask function, there is no possibility of network overhead or caching. The main reason should be: RESTful API can be slower than GraphQL if there exists over-fetching (fetching more data than wanted) or under-fetching (fetching less than wanted), and this situation often happens when there are multiple filters applied to the data being fetched.
However, in our case, **this API is designed to call local Python functions**, and **the returned data needs no querying**. Therefore, **there exists no possibility of over-fetching or under-fetching**. So RESTful API might be more straightforward than GraphQL, which uses multiple query to specify the result to return.

However, the choice of API technology also depends on other factors. For example, GraphQL can help minimize requests by its query flexibility.

• What are the benefits of using RESTful API? What are the benefits of using GraphQL?

Sudoku W/ Ease
Lab 1 Report

Yichen Guo yg2258@nyu.edu

06/14/23

Answer:
RESTful API Benefits:

1. Simplicity and ease of use: RESTful APIs are based on standard HTTP methods and utilize well-known principles such as URIs and status codes. This simplicity makes them easier to understand, implement, and consume. Developers familiar with HTTP can quickly grasp the concepts and start using RESTful APIs.
2. Scalability: RESTful APIs are stateless, meaning each request contains all the necessary information to process it. This statelessness allows them to be easily scaled horizontally by adding more servers, as requests can be distributed without relying on server-side sessions or shared state.
3. Compatibility: RESTful APIs are compatible with a wide range of clients and can be accessed from various platforms, including web browsers, mobile devices, and IoT devices. They are not tied to any specific programming language or framework, making them highly versatile.
4. Caching: RESTful APIs take advantage of HTTP caching mechanisms, allowing responses to be cached by intermediate proxies or clients. This reduces server load and improves performance by serving cached responses when applicable.

GraphQL Benefits:

1. Efficient data retrieval: GraphQL allows clients to specify exactly what data they need and retrieve it in a single request. This avoids over-fetching or under-fetching data, reducing unnecessary network traffic and improving performance. Clients have fine-grained control over the shape and structure of the response, which can be particularly beneficial in scenarios where bandwidth or latency is a concern.
2. **Flexibility and versioning**: GraphQL provides flexibility in data querying, allowing clients to request specific fields, relationships, and nested data structures. This flexibility enables frontend developers to request precisely what they need without requiring backend changes. It also simplifies versioning, as new fields or features can be added to the GraphQL schema without breaking existing clients.
3. Reduced round trips: With GraphQL, clients can retrieve multiple resources or related data in a single request. This reduces the number of round trips between the client and the server, improving efficiency and reducing network latency.
4. Development experience: GraphQL offers powerful tools for exploring the API schema, documenting the available data and operations, and providing strong typing. These features enhance the development experience, promote self-documenting APIs, and facilitate collaboration between frontend and backend developers.
5. **Real-time updates**: GraphQL supports **subscriptions**, which enable real-time updates and push notifications. This is particularly useful for applications that require live data, such as chat applications, collaborative tools, or real-time monitoring systems.

• Which one would you choose, RESTful API or GraphQL?
Answer:

Sudoku W/ Ease
Lab 1 Report

Yichen Guo yg2258@nyu.edu

06/14/23

I decided to use GraphQL. Though it is providing me a slower performance, GraphQL allowes more flexible querying, which can simplify my data retrieval from PostgreSQL, and live updates, which is very useful when solving the Sudoku takes longer than expected.