

This is a public copy of “y/why-graphql”, an internal pitch deck created to promote the use of GraphQL at Yelp for our web and mobile clients.

(References to “y/ . . .” are internal shortlinks)

**Published:** April 8th 2024

# Why you should use GraphQL!

[y/why-graphql](https://y/why-graphql)

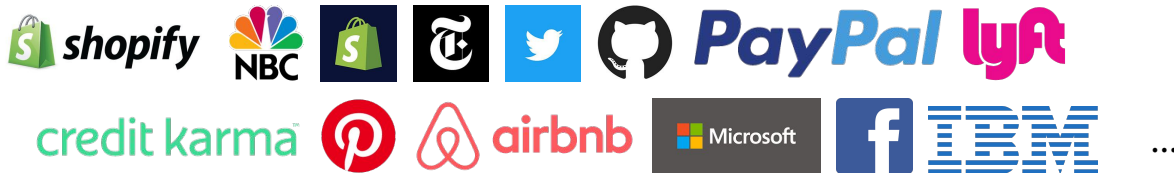
# What is it?

GraphQL is a query language that makes it super straightforward to fetch data in our React, iOS and Android apps **declaratively**.



<https://graphql.org/>

Used widely in industry by:



<https://graphql.org/users/>

The screenshot shows the GraphQL Playground web application. The browser address bar displays `graphql.yelpcorp.com`. The interface includes a tab labeled `GetBizName`, buttons for `PRETTIFY`, `HISTORY`, and `COPY CURL`, and a URL input field containing `https://graphql.yelpcorp.com/gql/graphql`. A central play button is positioned between the query and response areas.

**Query:**

```
1 query GetBizName($BizEncId: String) {  
2   business(encid: $BizEncId) {  
3     name  
4   }  
5 }
```

**Response:**

```
{  
  "data": {  
    "business": {  
      "name": "Mint Ice Mojito Kingdom"  
    }  
  },  
  "extensions": {}  
}
```

**Query Variables:**

```
1 {  
2   "BizEncId": "E24Bu-vMhVpueQpU8mpRPw"  
3 }
```

On the right side, there are vertical tabs for `DOCS` and `SCHEMA`. At the bottom right, a `TRACING` button is visible.

A large blue speech bubble is overlaid on the right side of the interface, containing the text: "Try it out now in the interactive GQL playground! 🖱️ [y/gql-playground](https://y/gql-playground)".

The Playground

# “Classic” REST Service (@ Yelp) Problems:

**Fetching data from multiple sources is hard.**

1. Developers have to **write and maintain significant amounts of data-fetching code** for every feature.
  - If you need data from n services, you must write (at minimum) n network requests
  - Duplicated 2x: for the Mobile API (for iOS/Android), and in BFFs (for Python templates for web)
  - Discovering existing endpoints for a type is hard, leading to duplicated overlapping public endpoints
2. Return objects from REST endpoints are **untyped**, meaning runtime code is less safe.
3. Slower iteration time due to the complexities of fetching data mean **features take longer to ship**

See [y/cep980](https://www.yelp.com/eng/cep980) for a full breakdown.

**How Does GraphQL Help?**

## 1. **Developer Velocity** (Improving Time to Market)

Overall, we believe GraphQL will significantly cut down on the time developers need to spend writing code related to fetching data. This means **faster iteration cycles and product releases**.

~ y/graphql-docs, 2019

## 1. Developer Velocity (Improving Time to Market)

Overall, we believe GraphQL will *win* on the time to market for building data. This means *winning*.

~ y/graphql

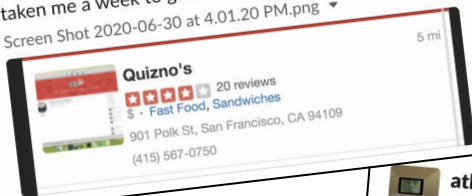


shecht 4:00 PM

can i just say gql/gss+++++

i'm rebuilding the msite search page and am building out the search results right now. all i have is the biz id and the distance string, and i was able to build all of this in 2 days, and the second day has just been working with css. this easily would have taken me a week to get all the calls together to get all this data

Screen Shot 2020-06-30 at 4.01.20 PM.png



ath 3:33 PM

@taras



taras

agree

every time we add to biz\_info\_pages, its clearer and clearer why GraphQL makes all this so much easier

Tuesday, May 12th

#graphql

41 | 0 | onpoint: @gql-group | y/graphql-docs | y/graphql-roadmap | GraphQL

Friday, October 25th



jackguy 2:54 PM

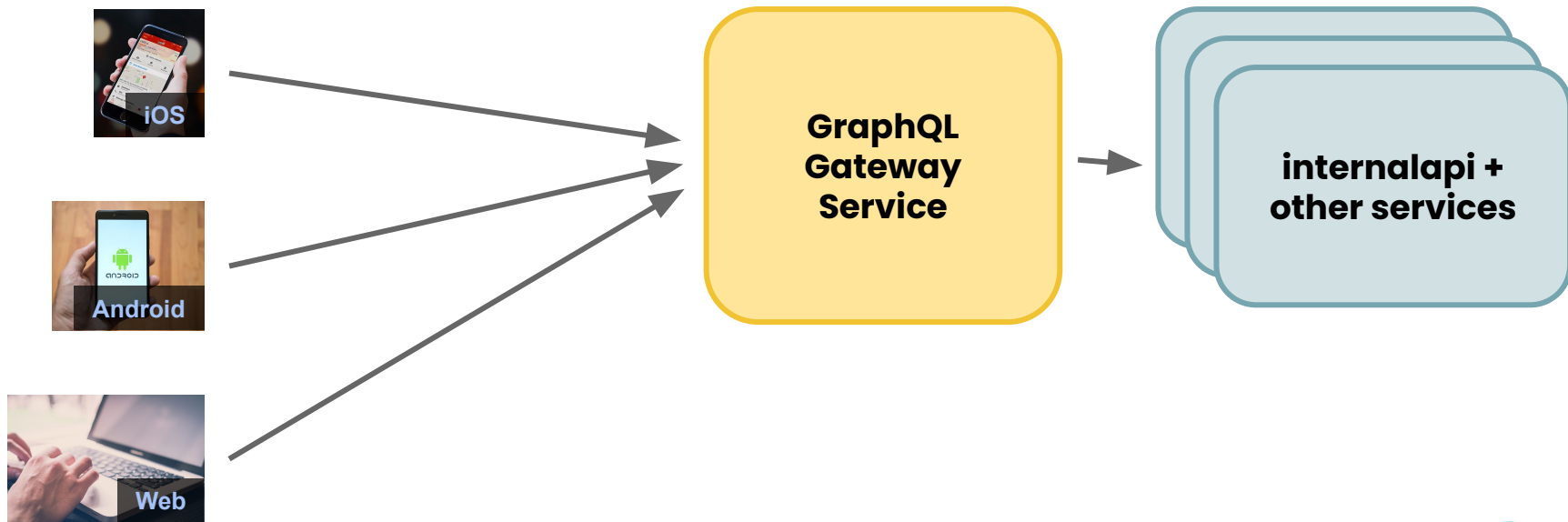
I'm sure this is old news at this point but I just needed to get a new field for yrc-ynra and not having to go through a props workflow was so amazing

It took me 2 minutes instead of 2 days

Keep up the good work soldiers



## 2. Unified API Server for Consumer, Biz, Web, iOS, Android, BizApp



*Shared reuse of backend logic for all our apps!*

**GGS can be called from anywhere** - the web, iOS, Android and BizApp. Your team can write **shared backend logic** once and reuse it on all our client platforms!

### 3. Simplified Component API + Usage (Developer Velocity)

Example:



Webcore / WEBCORE-7904

## Render the Consumer WWW React Header using GraphQL

Description

Goal

To render the Consumer WWW React Header on high traffic pages using GraphQL (as an experiment option)



Status Quo

```
<ConsumerHeader
  holidayModifier={props.holidayModifier}
  userData={props.userData}
  bizSiteURL={props.bizSiteURL}
  currentURL={props.currentURL}
  logoutCsrfToken={props.logoutCsrfToken}
  initialLocationDisplayVal={props.initialLocationDisplayVal}
  initialHiddenLocationInputName={props.initialHiddenLocationInputName}
  initialHiddenLocationInputVal={props.initialHiddenLocationInputVal}
  leftLinksData={props.leftLinksData}
  rightLinksData={props.rightLinksData}
  initialIsCurrentLocation={false}
  initialSearchDisplayVal={' '}
  suggestKey={null}
  suggestSiteRepParams={null}
  uniqueRequestId={props.uniqueRequestId}
/>
```

The Glorious Future

Ultimately, we want devs to be able to write something like this to render the header:

```
<ConsumerHeader />
```



*No data props required!*

**Components can fetch their own data props! Wow! No need to write props fetching code!**

## 4. Simplified Data Fetching (Developer Experience + Happiness)

*Reduced boilerplate (e.g. setting up multiple clientlibs, asyncio loops)*

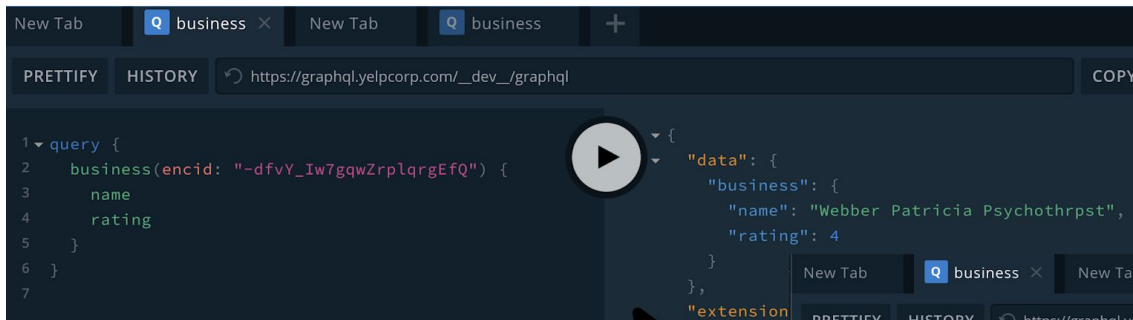
```
> homepage / views / home.py
241 swaggerpy_client = get_internal_api_client()
242 messaging_client = get_messaging_client()
243
244 messages_unread_count_future = loop.run_in_executor(
245     executor=None,
246     func=messaging_client.inbox.unread_message_count_for_user(
247         user_id=unauth_user_id,
248     ).result
249 ) if unauth_user_id else None
250
251 notification_unread_count_future = loop.run_in_executor(
252     executor=None,
253     func=bravado_client.user_alerts.get_user_alerts_unread_count_for_user(
254         user_id=unauth_user_id
255     ).result
256 ) if unauth_user_id else None
257
258 geoup_location_future = asyncio.ensure_future(fetch_geoup_location(
259     bravado_client=bravado_client,
260     remote_addr=get_remote_addr(request),
261     locale=locale,
262 ))
263
264 fetch_recent_locations_future = asyncio.ensure_future(fetch_recent_locations(
265     request=request,
266     user_id=unauth_user_id,
267     bravado_client=bravado_client,
268     geolocator_client=geolocator_client,
269     locale=locale,
270 ))
271
272 hero_user_future = loop.run_in_executor(executor=None, func=swaggerpy_client.user.retrieve_users_by_id(
273     ids=str(decid(hero_photo_owner_id)),
274     locale=locale,
275     omit_private_info=True,
276 ).result) if hero_photo_owner_id else None
277
278 hero_biz_future = loop.run_in_executor(executor=None, func=swaggerpy_client.business.list_by_ids_v4(
279     business_ids=str(decid(hero_biz_id)),
280 ).result) if hero_biz_id else None
281
282 logged_in_user_future = loop.run_in_executor(executor=None, func=functools.partial(
283     homepage.user.retrieve_users_from_request,
284     request=request,
285 )) if unauth_user_id else None
286
```

gondola-homepage / src / components / RecentActivityGraphQL / RecentActivityGraphQL.js

Jack Bug fixes for homepage activity feed 19 days ago Show history

```
56
57 query GetRecentActivityFeedData(
58   $feed: PersonalActivityFeedType!
59   $nearbyFeed: Boolean!
60   $resultsPerPage: Int!
61   $reactionsSourceFlow: String!
62   $after: String
63 ) {
64   ...BusinessPhotoFeedItemRootFields
65   loggedInUser {
66     encid
67     personalActivityFeed(feed: $feed, first: $resultsPerPage, after: $after) @skip(if: $nearbyFeed) {
68       edges {
69         node {
70           ...HomeActivityFeedGroupFields
71         }
72       }
73       pageInfo {
74         endCursor
75         hasNextPage
76       }
77     }
78   }
79   nearbyActivityFeed(first: $resultsPerPage, after: $after) @include(if: $nearbyFeed) {
80     edges {
81       node {
82         ...HomeActivityFeedGroupFields
83       }
84     }
85     pageInfo {
86       endCursor
87       hasNextPage
88     }
89   }
90 }
91
92
```

## 5. Frontend Changes without Backend Changes (Improving **Time to Market** through quicker iteration times)



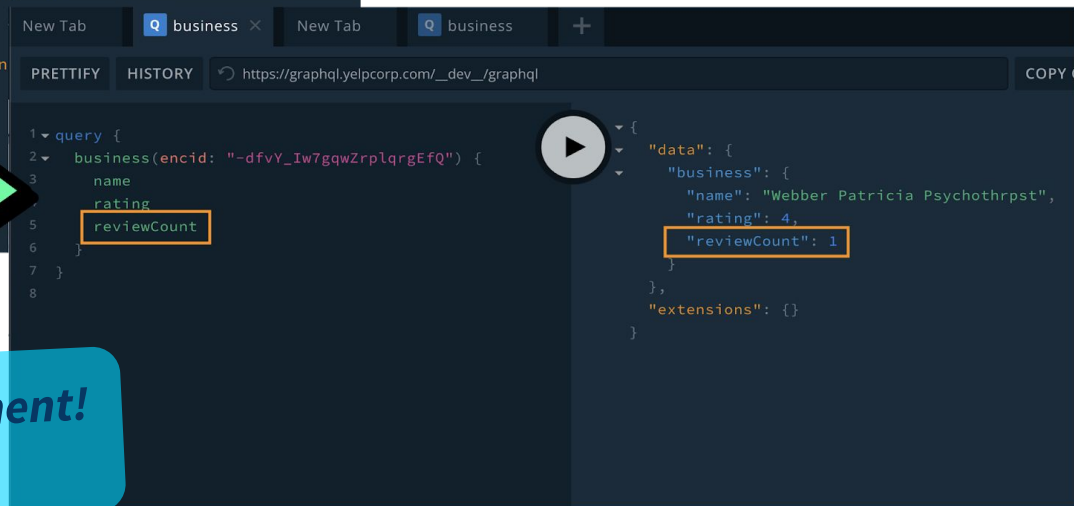
A screenshot of the GraphQL Playground interface. The top bar shows two tabs labeled 'business' and a search icon. The URL bar contains 'https://graphql.yelpcorp.com/\_\_dev\_\_/graphql'. The left pane shows a query editor with the following code:

```
1 query {  
2   business(encid: "-dfvY_Iw7gqwZrplrgEfQ") {  
3     name  
4     rating  
5   }  
6 }  
7
```

The right pane shows the JSON response:

```
{  
  "data": {  
    "business": {  
      "name": "Webber Patricia Psychothrpst",  
      "rating": 4  
    }  
  },  
  "extensions": {}  
}
```

A large green arrow points from the 'reviewCount' field in the query editor to the 'reviewCount' field in the response.



A screenshot of the GraphQL Playground interface, similar to the one above, but with the query updated to include 'reviewCount'. The left pane shows the query:

```
1 query {  
2   business(encid: "-dfvY_Iw7gqwZrplrgEfQ") {  
3     name  
4     rating  
5     reviewCount  
6   }  
7 }  
8
```

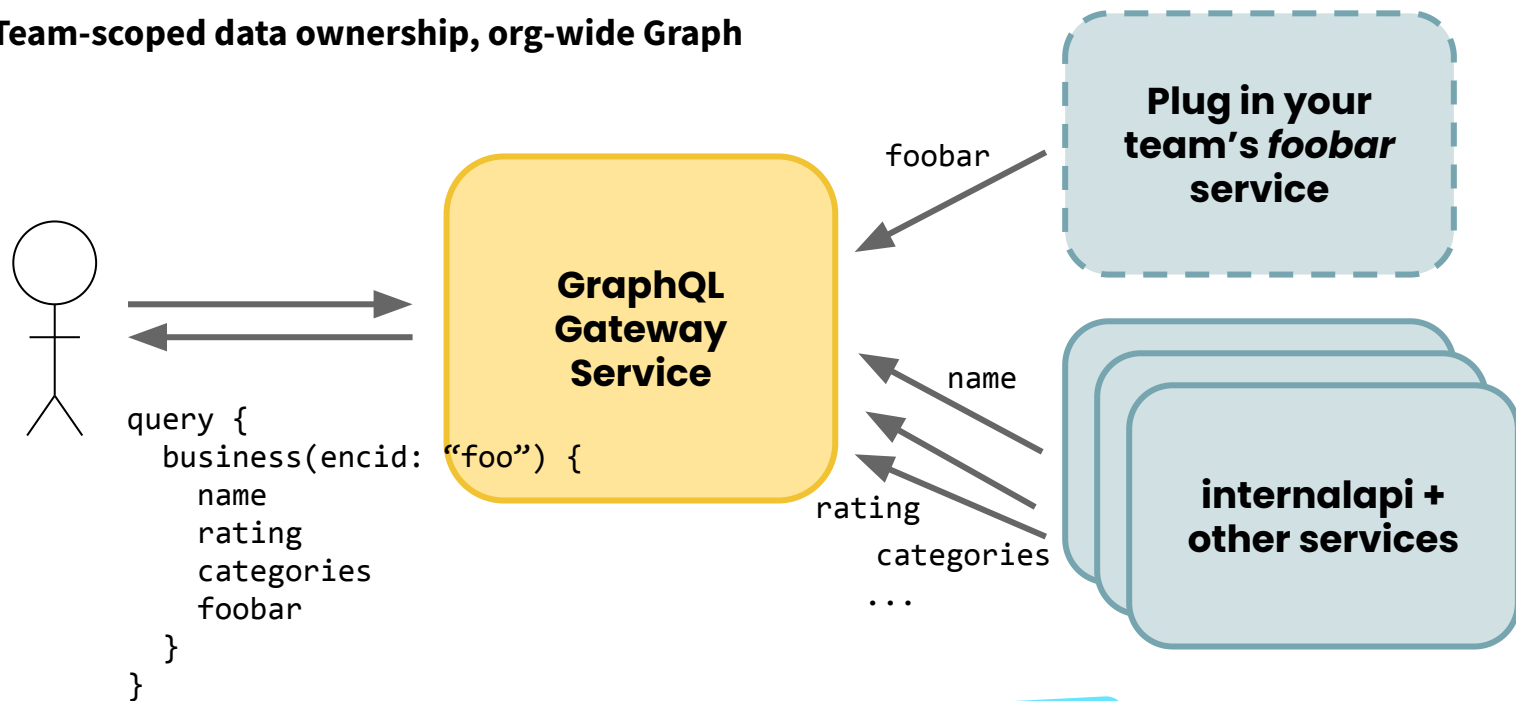
The right pane shows the updated JSON response:

```
{  
  "data": {  
    "business": {  
      "name": "Webber Patricia Psychothrpst",  
      "rating": 4,  
      "reviewCount": 1  
    }  
  },  
  "extensions": {}  
}
```

The 'reviewCount' field in both the query and the response is highlighted with a yellow box.

***Just update the query in the component!***  
*No need to push a backend service!*

## 6. Team-scoped data ownership, org-wide Graph



***Only manage the data your team owns!***


Your page can fetch all the data it needs in **one** GQL query (including other team's data) without you having to worry about *how* it's all fetched. Just supply the encid of the entities you want - **GGs already knows how to fetch everything else!**


## 7. Zero-risk of breaking clients with breaking changes


**Enforced safety for API changes.** Removed a field? Our Github PR bot prevents you deploying breaking changes that would break clients!

svc-jenkins commented 8 hours ago • edited



Member

 **Schema Check Bot**


 Found 4 schema changes, including 1 breaking changes and 0 unsupported changes

 **Schema Changes**

► Show all schema changes


 **Breaking Changes Detected** 

Removing fields from the schema **will break pages** containing queries that use those fields. To learn how to safely ship breaking changes, follow the "Breaking Changes" section in [y/schema-checks](#).

 **Do not ship this PR!** The table below shows all documents **used in production within the last 2 weeks** that would be broken by this PR:

▼ Show all affected documents

Field	Production Usage
<code>InvoiceCapabilitiesCohort.isInPilot</code>	Used by the following documents: <ul style="list-style-type: none"><li><code>aa3794f71b1fd571fbc5c3afa06ac9d3daae1a7e5e3c3b6f6eb55f255ceab3d7</code> (last executed less than a minute ago)</li></ul>

 Use the CLI to view more information about these documents:

```
$ graphql-gateway-document-store find-documents --field "[insert field here]" -a "$(date --date="2 weeks ago")"
```

## Case Study: “From This Business” product migration to GraphQL on Mobile

### FTB GQL migration: results

get from this business info.graphql

- **280~310 ms** response time (~2.5x faster)

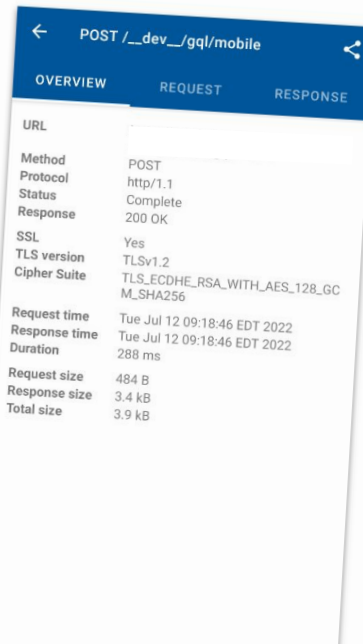
View and API logic separation

Faster development cycle

Tech Spec

#proj-ftb-modernization

Biz: Android Sync



POST /__dev__/gql/mobile	
OVERVIEW	REQUEST
RESPONSE	
URL	
Method	POST
Protocol	http/1.1
Status	Complete
Response	200 OK
SSL	Yes
TLS version	TLSv1.2
Cipher Suite	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
Request time	Tue Jul 12 09:18:46 EDT 2022
Response time	Tue Jul 12 09:18:46 EDT 2022
Duration	288 ms
Request size	484 B
Response size	3.4 kB
Total size	3.9 kB

👉 <link to case study doc>

# Next steps

## What's in it for my team?

- Faster iteration cycles and product releases
- More delightful developer experience

## We want you to try it out!

GraphQL is fully available for all developers at Yelp! If you're interested in seeing how GraphQL can save you time on your upcoming project, **please reach out in #graphql!**

## Where can I learn more?

 [y/graphql-docs](#) 