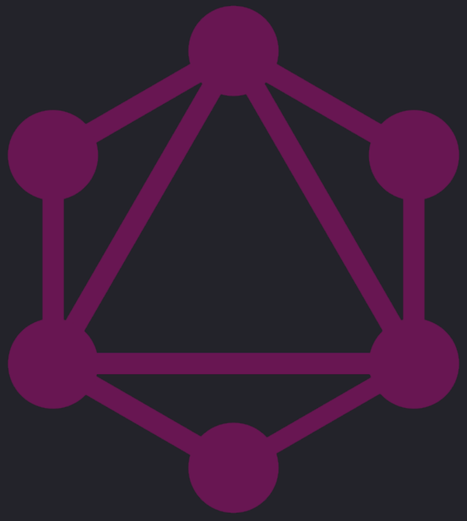
Getting started with GraphQL

In an enterprise

Shruti Kapoor

@shrutikapoor08



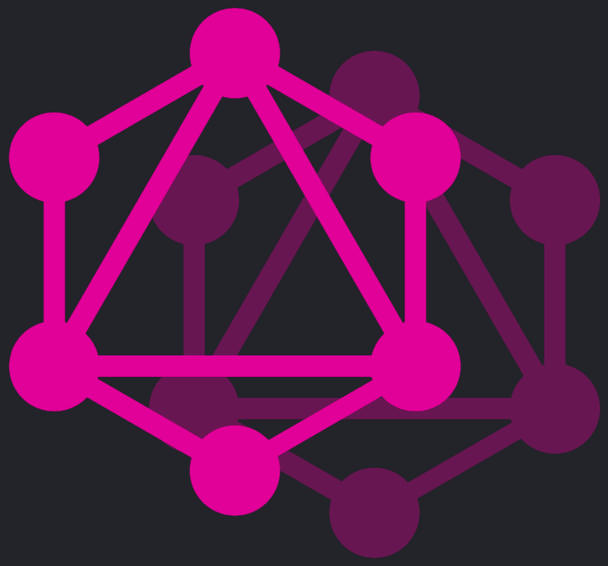
Outline

Why GraphQL?

Our story of adopting GraphQL in enterprise

Lessons we learnt

How you can get started



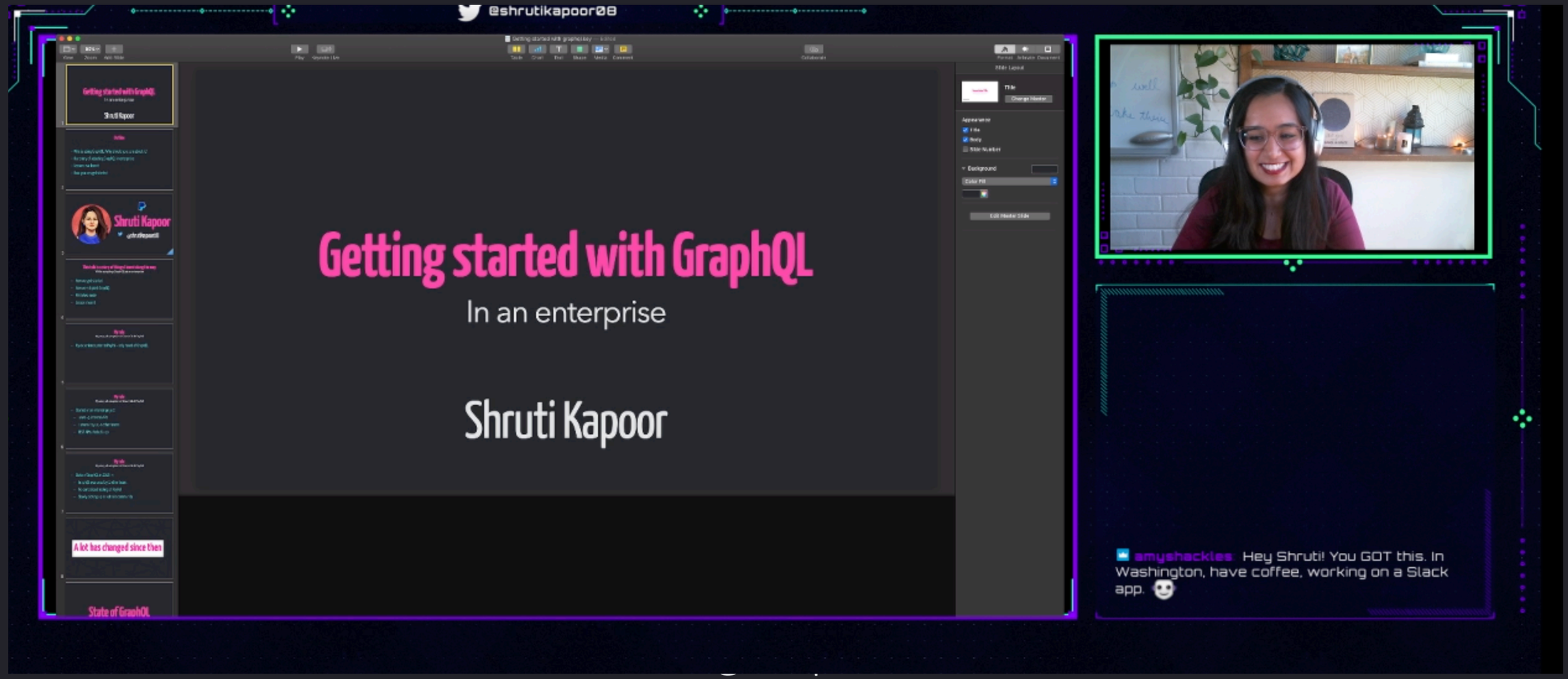
Shruti Kapoor

@shrutikapoor08

@shrutikapoor08

twitch.tv/shrutikapoor

twitch.tv/shrutikapoor



@shrutikapoor08

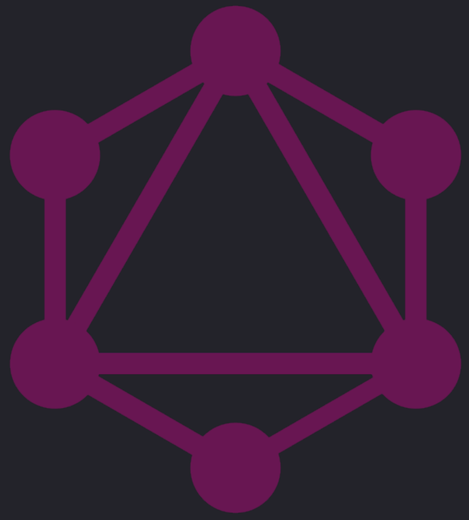
Me before PayPal

My role

Started in an internal project

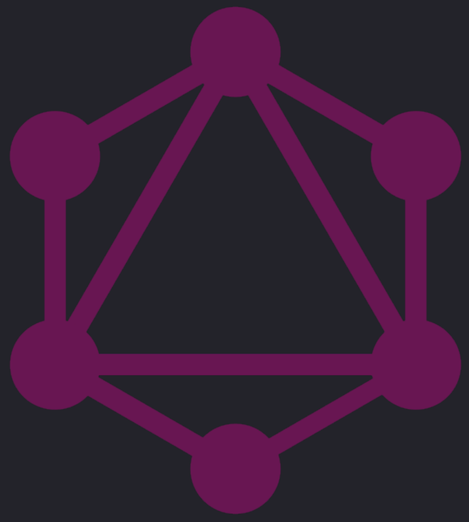
<5 internal APIs

REST APIs, NodeJS app



State of GraphQL in 2018

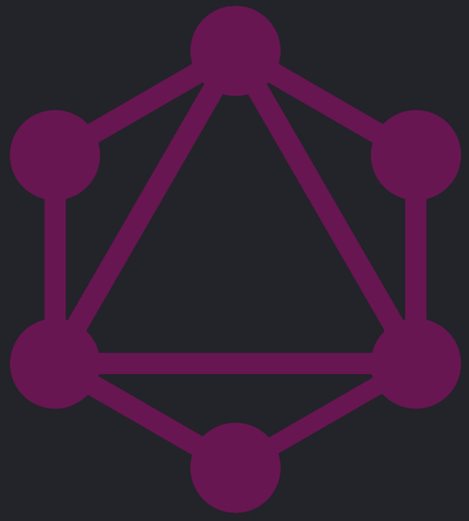
Checkout was the trailblazer



@shrutikapoor08

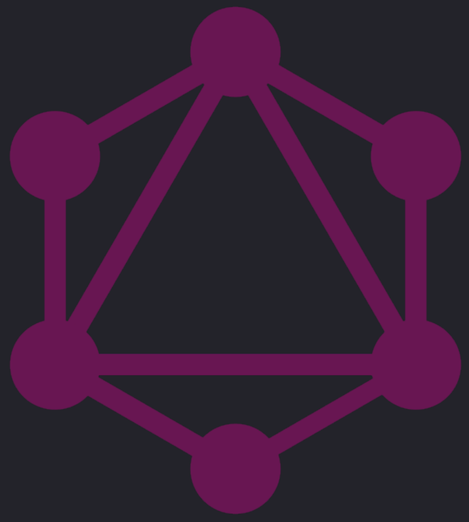
State of GraphQL in 2018

No central support initially

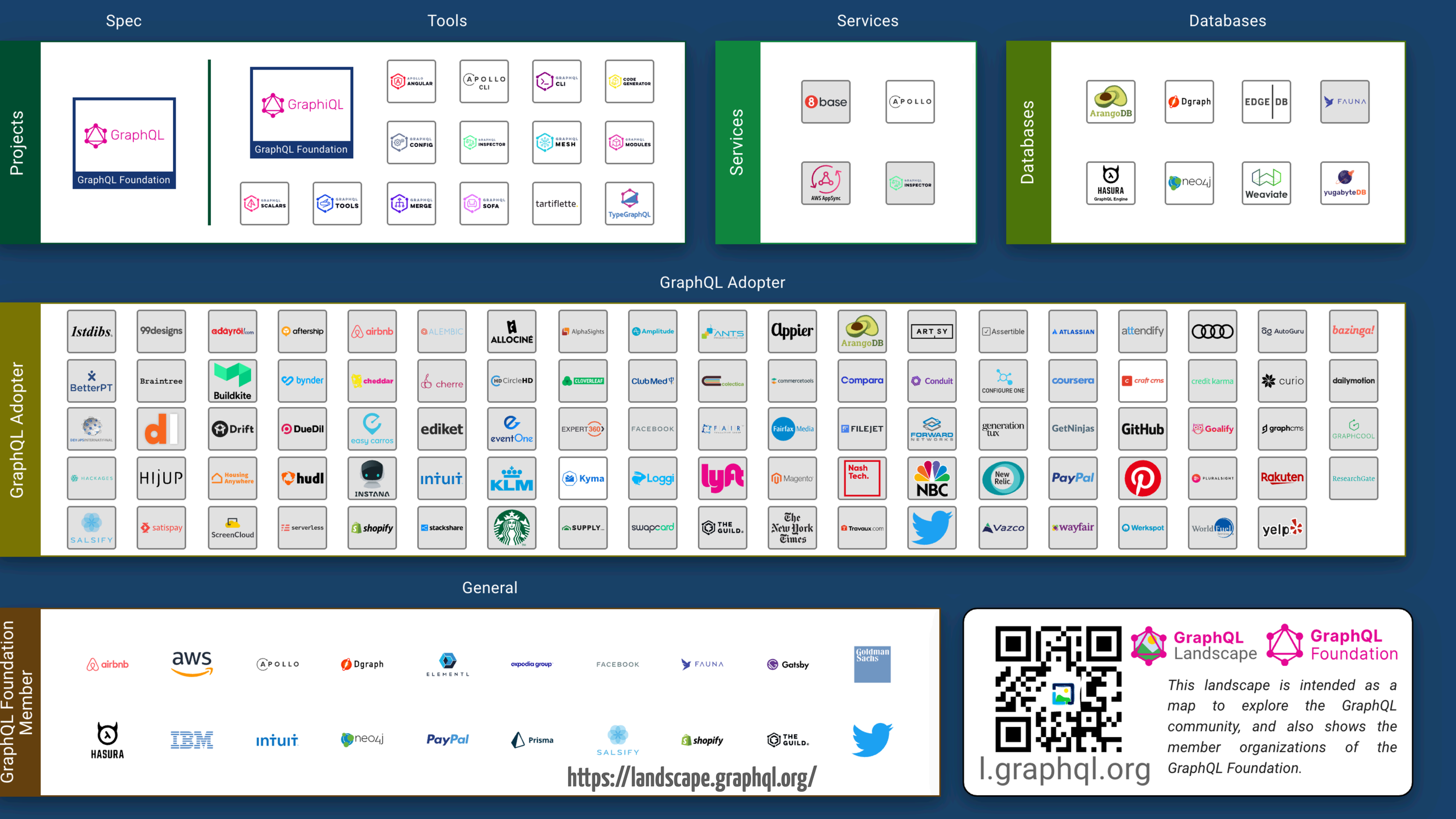


State of GraphQL in 2018

Early community adoption



A lot has changed since then


Spec
Tools
Services
Databases
Projects
GraphQL
GraphQL Foundation
GraphiQL
GraphQL Foundation
Services
Databases
GraphQL Adopter
GraphQL Adopter
General
GraphQL Foundation Member
https://landscape.graphql.org/
l.graphql.org
GraphQL Landscape
GraphQL Foundation
This landscape is intended as a map to explore the GraphQL community, and also shows the member organizations of the GraphQL Foundation.

graphql / graphql-js

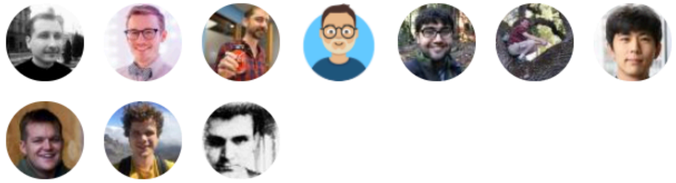
Unwatch 412 | Unstar 17.3k | Fork 1.8k

Code | Issues 115 | Pull requests 57 | Actions | Projects 1 | Wiki | Security | Insights

Used by 437k

+ 437,431

Contributors 165



+ 154 contributors

https://github.com/graphql/graphql-js

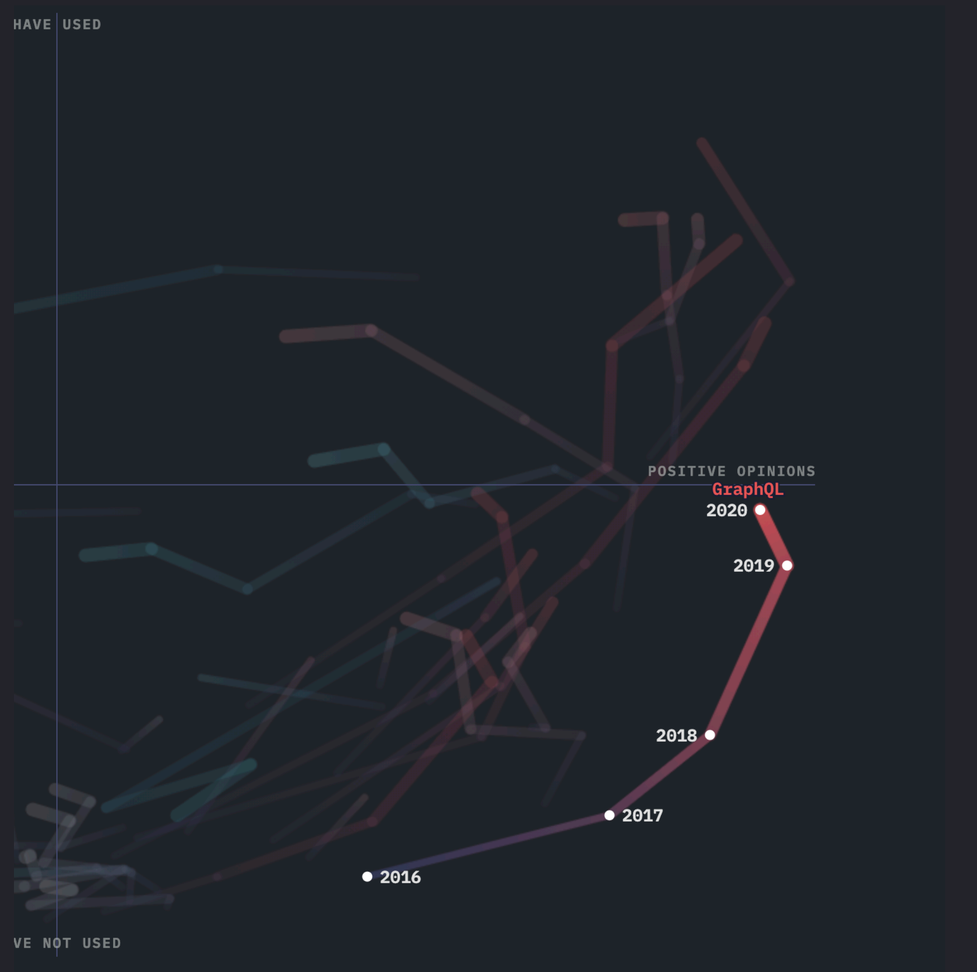
@shrutikapoor08

Highest Interest

Awarded to the technology developers are most interested in learning once they are aware of it.

GraphQL

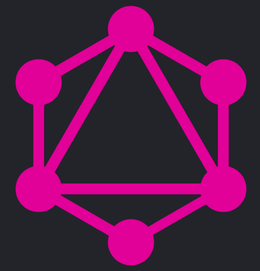
GraphQL continues to be the one thing developers want to learn more about with an interest ratio of **86%**, as soon as they can finally find the time.



https://2020.stateofjs.com/en-US/awards/

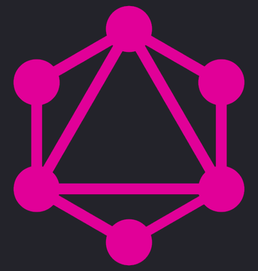
@shrutikapoor08





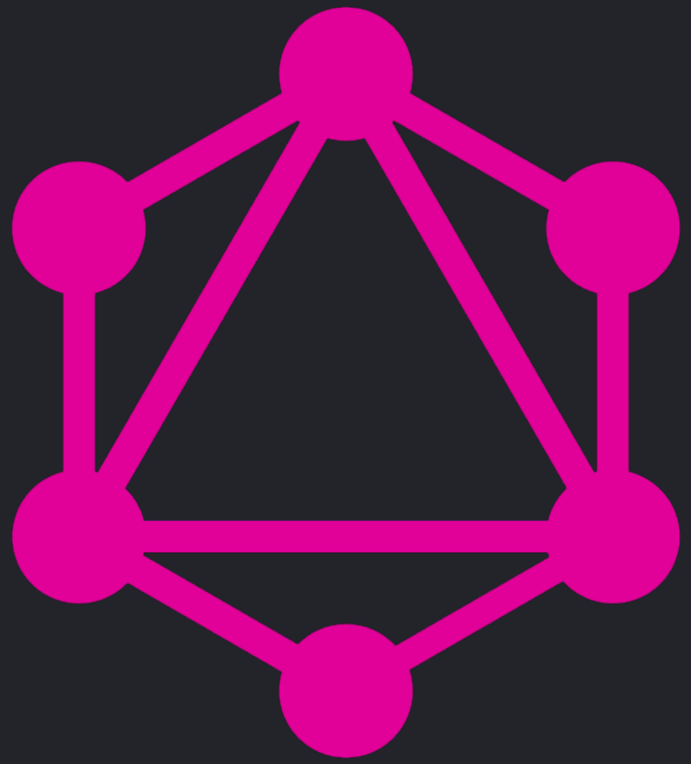
~50 apps in production using GraphQL
Braintree public API
> 500 members in GraphQL community





Built internal tools
Provide Infra support
Provide training materials
Default pattern for building UI
Used across the board


GRAPH A WHA?
MAKES NO SENSE
memegenerator.ne



GraphQL Terminology

@shrutikapoor08

Graph | QL

@shrutikapoor08

Slide by Mark Stuart, PayPal

Graph

QL

Graph because data is represented as a graph.

QL for query language, not a SQL like technology

Slide by Mark Stuart, PayPal

A query language for APIs

@shrutikapoor08

> A query language for APIs

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools.

> A query language for APIs

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, **gives clients the power to ask for exactly what they need and nothing more,** makes it easier to evolve APIs over time, and enables powerful developer tools.

Getting data with REST

/customers/account

Getting data with REST

/customers/account

/auth

/customers/user/{id}

/customers/account-details

/customers/user/{id}/notifications

Getting data with REST

/customers/account

/auth

/customers/user/{id}

/customers/account-details

/customers/user/{id}/notifications

Getting data with REST

/customers/account

/auth

/customers/user/{id}

/customers/account-details

/customers/user/{id}/notifications

Getting data with REST

/customers/account

/auth

/customers/user/{id}

/customers/account-details

/customers/user/{id}/notifications

Getting data with REST

/customers/account

/auth

/customers/user/{id}

/customers/account-details

/customers/user/{id}/notifications

Getting data with GraphQL

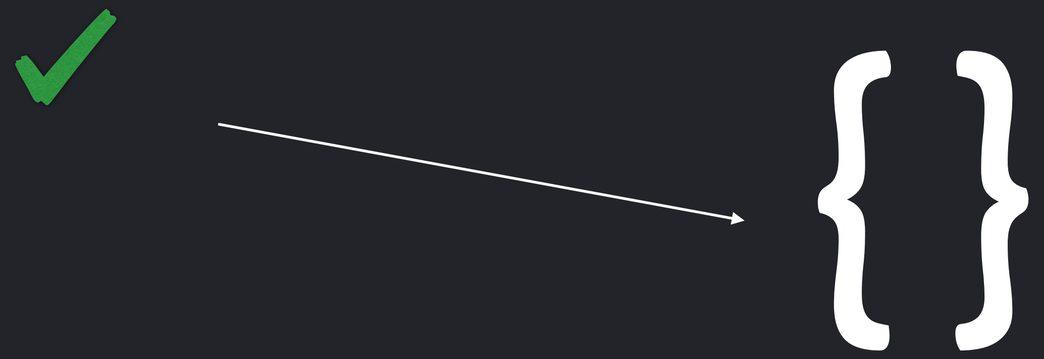
paypal.com/graphql

Getting data with GraphQL

paypal.com/graphql

Getting data with GraphQL

paypal.com/graphql



Schema
Queries
Mutations
Resolvers

Schema

a set of types which completely describe the set of possible data you can query on that service

Schema

```
type Character {
  name: String!
  appearsIn: [Episode!]!
}
```

```
type Starship {
  id: ID!
  name: String!
  length(unit: LengthUnit = METER): Float
}
```

Query

```
query {
  hero {
    name
  }
  droid(id: "2000") {
    name
  }
}
```

Query

```
query {
  hero {
    name
  }
  droid(id: "2000") {
    name
  }
}
```

```
{
  "data": {
    "hero": {
      "name": "R2-D2"
    },
    "droid": {
      "name": "C-3PO"
    }
  }
}
```

Query GET

Query	GET
Mutation	POST / DELETE

Resolvers

A function that “resolves” what data should be given back for a field

Every field has a resolver.

Resolvers

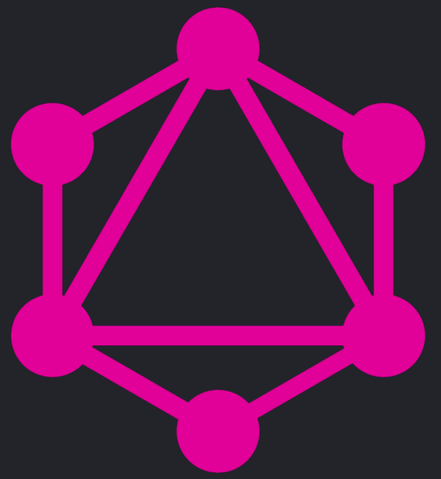
```
type User {
  # A User's first/given name (Ex: 'Marty')
  givenName: String

  # A User's last/family name. (Ex: 'McFly')
  familyName: String

  """
  A User's addresses.

  By default, returns all addresses.
  Optionally, passing a 'type' returns addresses of that type.
  """
  addresses(type: AddressType): [Address]
}
```

Resolvers

```
type User {
  # A User's first/given name (Ex: 'Marty')
  givenName: String

  # A User's last/family name. (Ex: 'McFly')
  familyName: String

  """
  A User's addresses.

  By default, returns all addresses.
  Optionally, passing a 'type' returns addresses of that type.
  """
  addresses(type: AddressType): [Address]
}
```

```
export default {
  Query: {
    user: async (root, args, { req }) => await getUser(req)
  },
  User: {
    givenName: ({ givenName }) => givenName,

    // By the way, these are optional ^^.
    // graphql-js has default resolvers
    familyName: ({ familyName }) => familyName,

    addresses: ({ addresses }, { type }) => {
      if (type) {
        return addresses.filter(address => address.type === type);
      }

      return addresses;
    }
  }
}
```

Features

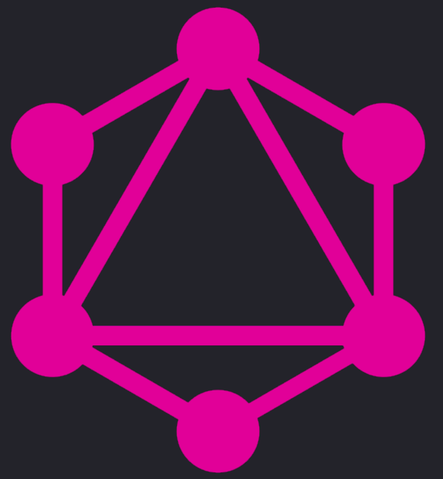


An open source specification

Clients ask for exactly the data they need, no more, no less.

Clients specify the structure of data they need

Features



No versioning

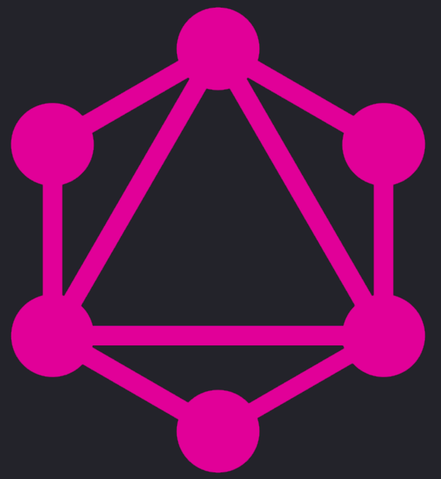
```
type Film {
  title: String
  episode: Int
  releaseDate: String
  openingCrawl: String
-  director: String
   directedBy: Person
}

type Person {
  name: String
  directed: [Film]
  actedIn: [Film]
}
```

```
type Film {
  title: String
  episode: Int
  releaseDate: String
  openingCrawl: String
+  director: String @deprecated
   directedBy: Person
}

type Person {
  name: String
  directed: [Film]
  actedIn: [Film]

}
```

```
type Film {
  title: String
  episode: Int
  releaseDate: String



}
```

```
type Film {
  title: String
  episode: Int
  releaseDate: String
+  openingCrawl: String


}
```

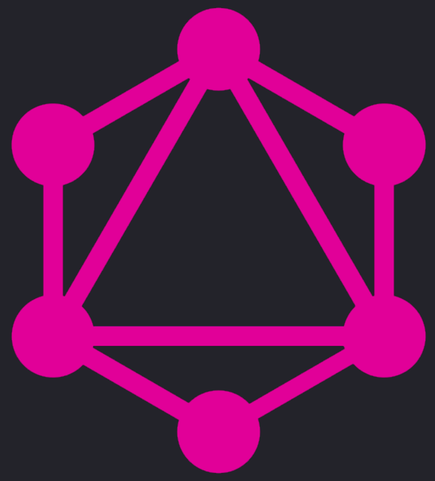
Features



Schemas: a contract between client and server

Developer UX: tools to ease learning and integration

Reasons to adopt



Our challenges at the time

Overfetched Data

Multiple round trips

Outdated clients

Inconsistent integration experience

The right candidate would provide...

Uniform experience

Ease of integration across board

Detailed analytics

A common conversation...

“Need to add data”

Versioned APIs?

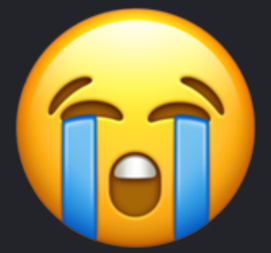
“Need to add data”

“Need to add data”

Versioned APIs?

breaking changes?

updates not delivered without reintegration?



New endpoint?

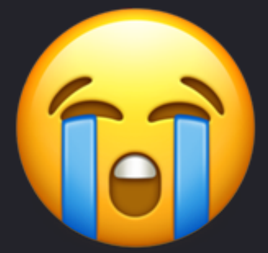
“Need to ask for more data”

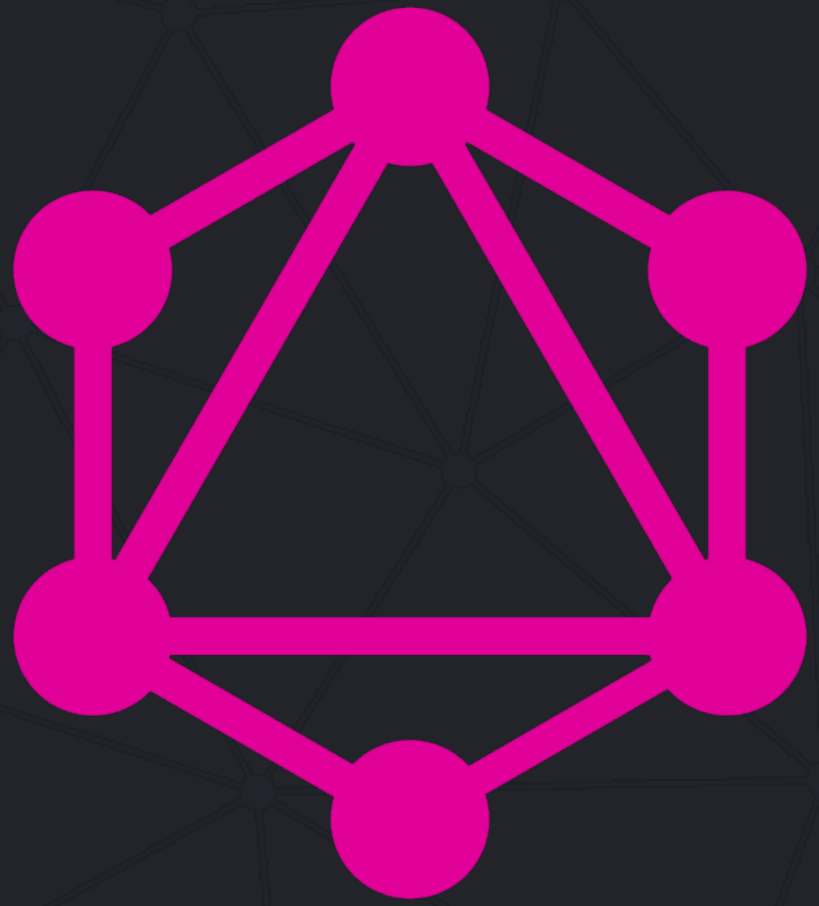
“Need to ask for more data”

New endpoint?

Add query params?

Add to existing endpoint?

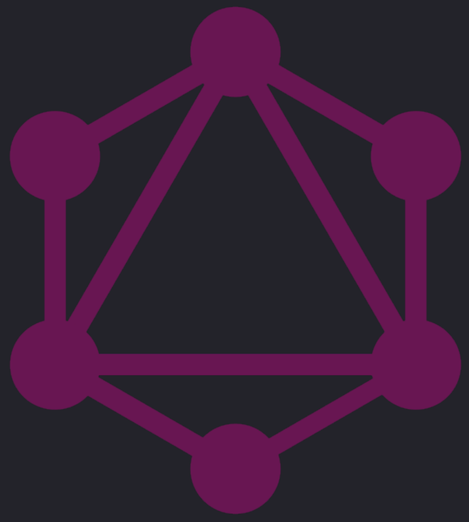




@shrutikapoor08

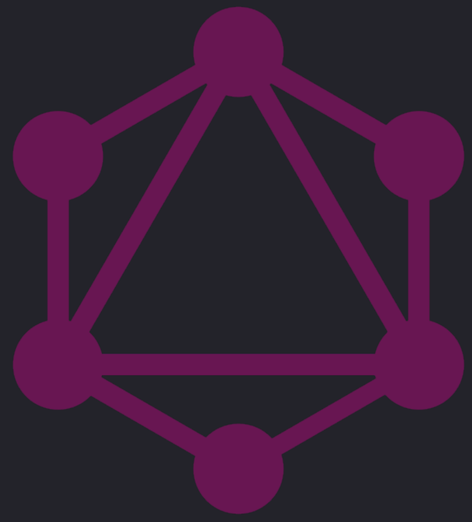
Why GraphQL?

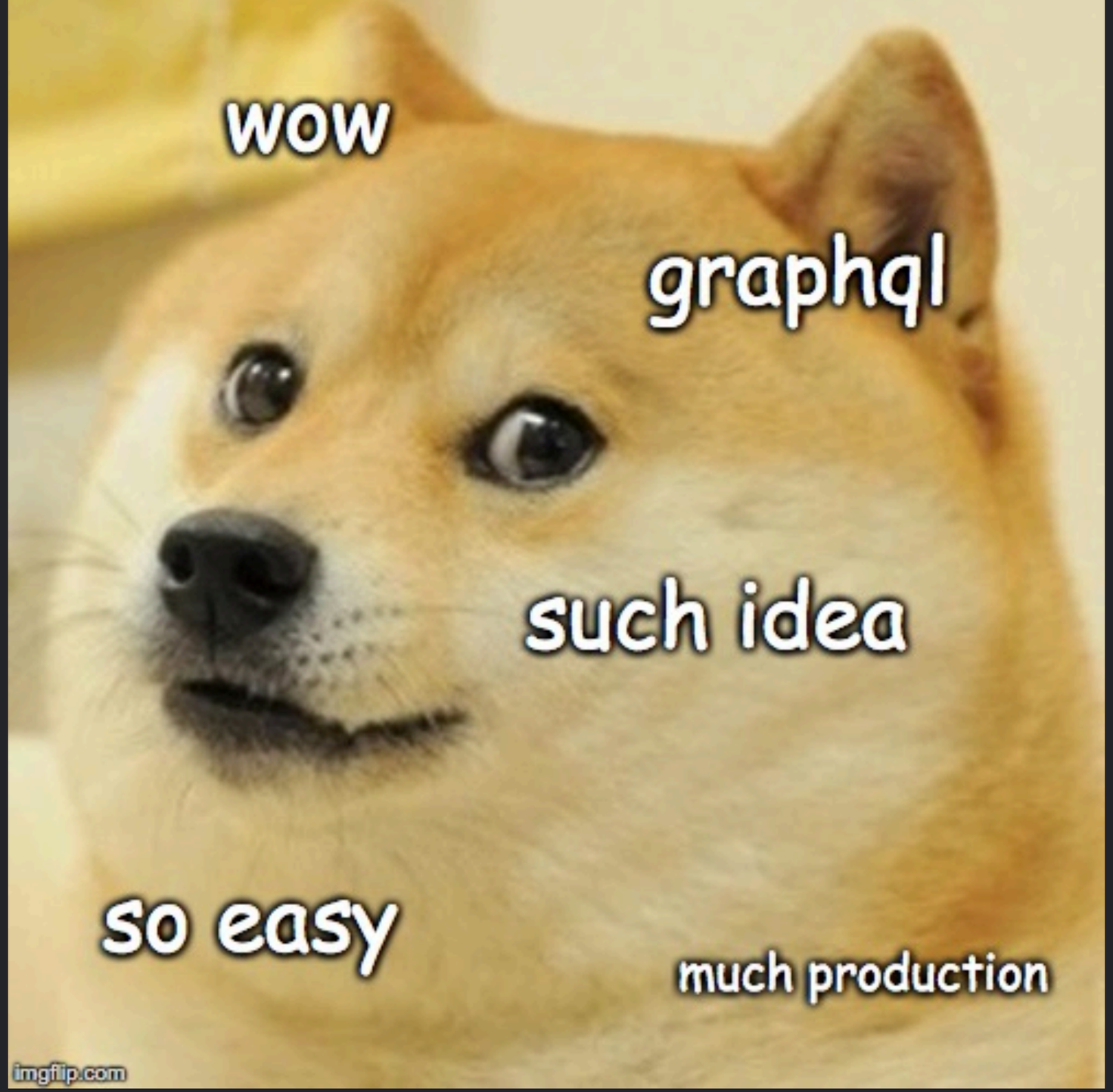
One endpoint
No more versioned APIs.
Easy to add and deprecate fields
Small payloads

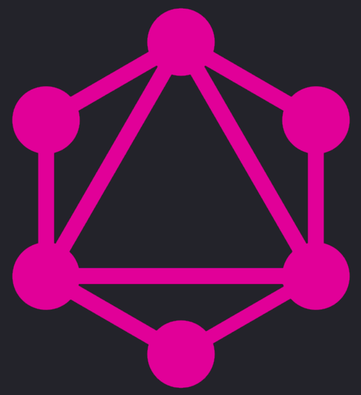


As a developer

Developer tooling
Collaboration across teams
Schema drives independent work



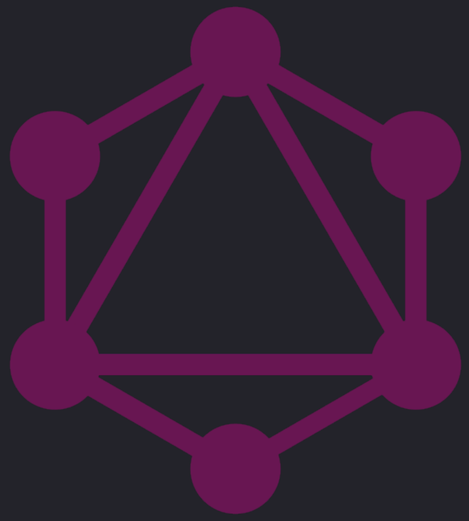

wow
graphql
such idea
so easy
much production
imgflip.com



How to adopt GraphQL

@shrutikapoor08

GraphQL wrapper over REST API

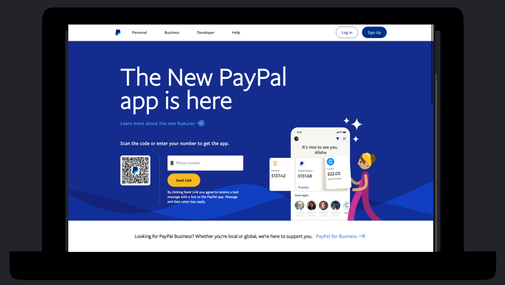


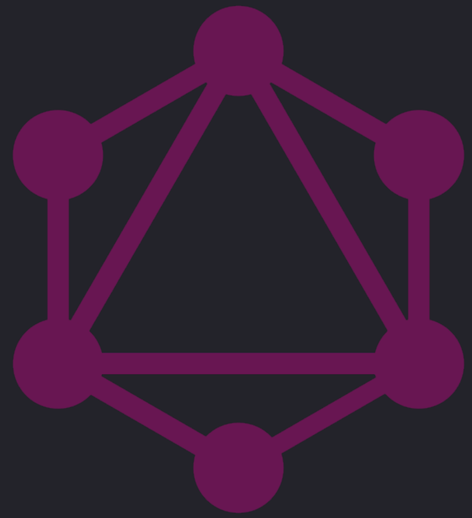
@shrutikapoor08

REST API

REST API

REST API



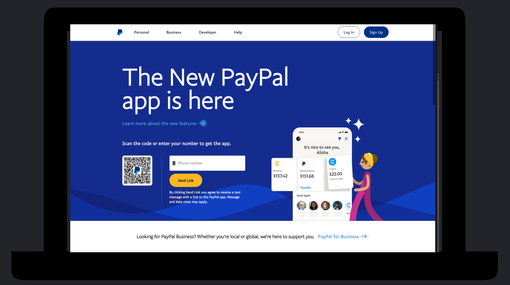


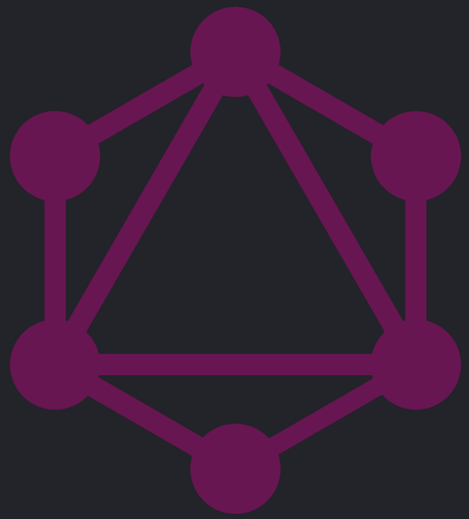
REST API

REST API

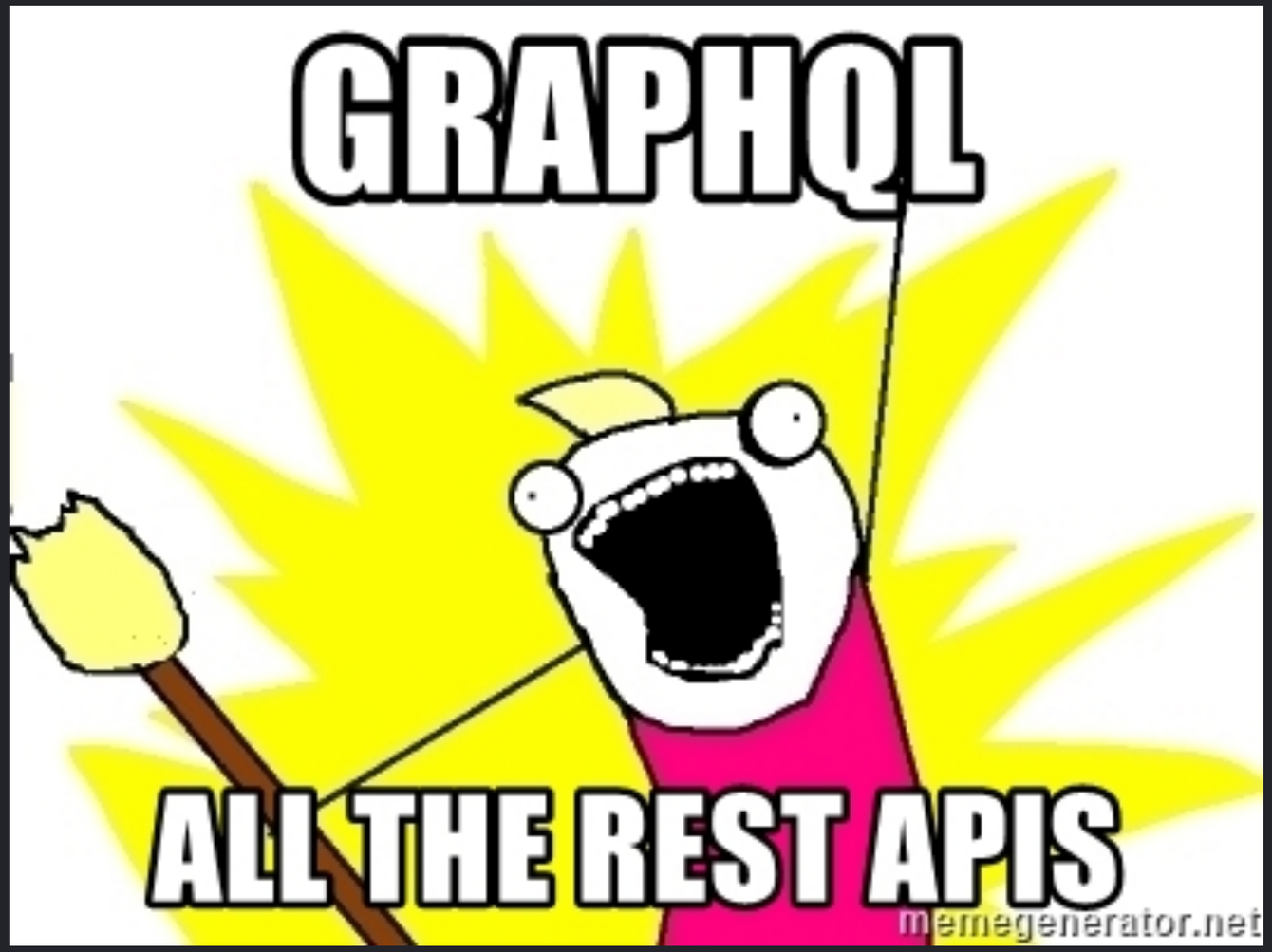
REST API

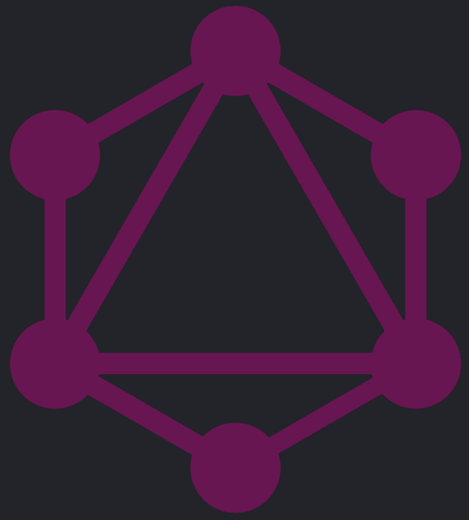
GraphQL API



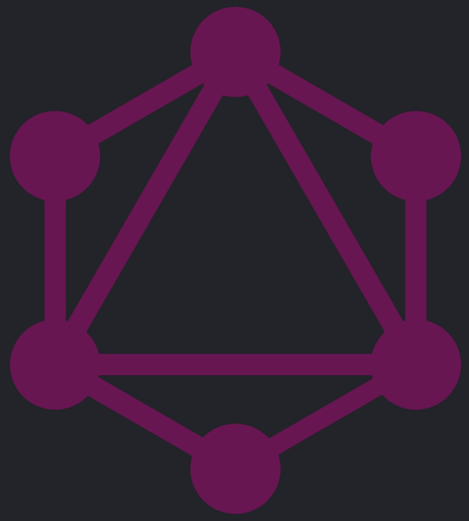


GraphQL wrapper over REST API





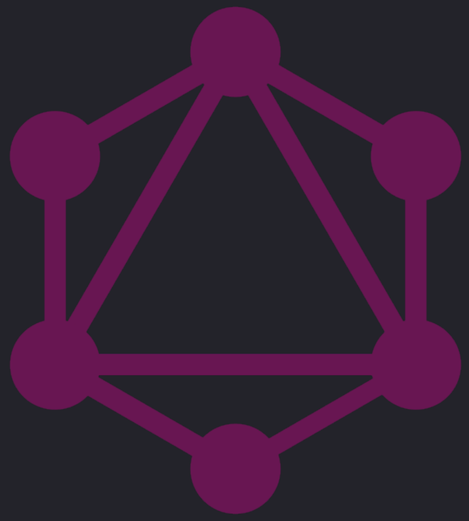
REST wrapper over GraphQL API



@shrutikapoor08

REST wrapper over GraphQL API

for clients that are tied to REST APIs



@shrutikapoor08

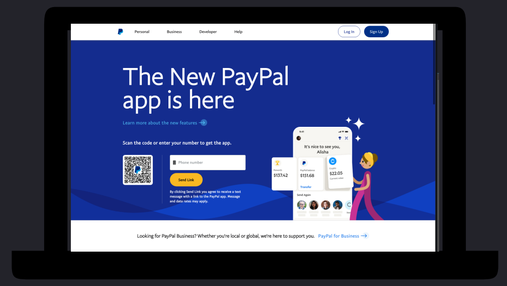
REST API

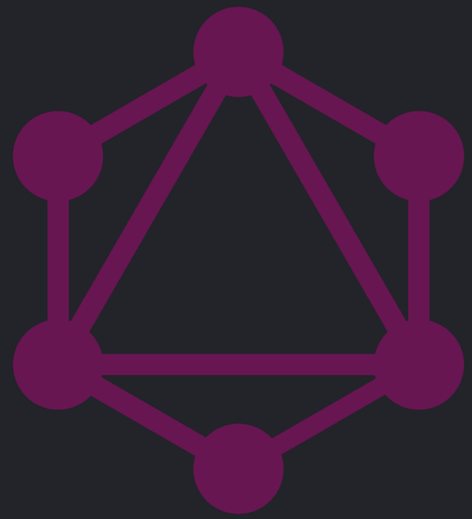
REST API

REST API

GraphQL API

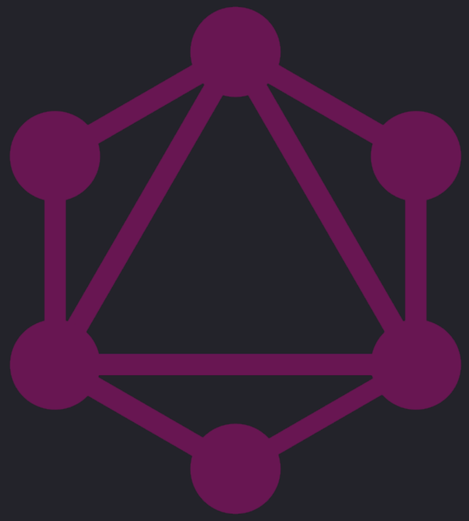
REST API





@shrutikapoor08

GraphQL and REST co-exist



@shrutikapoor08

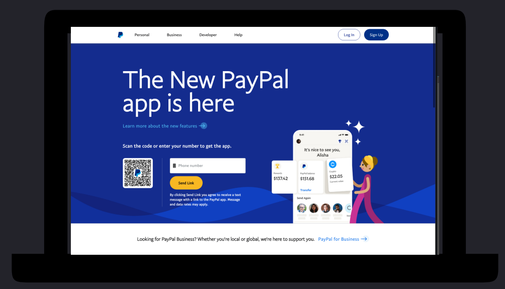
REST API

REST API

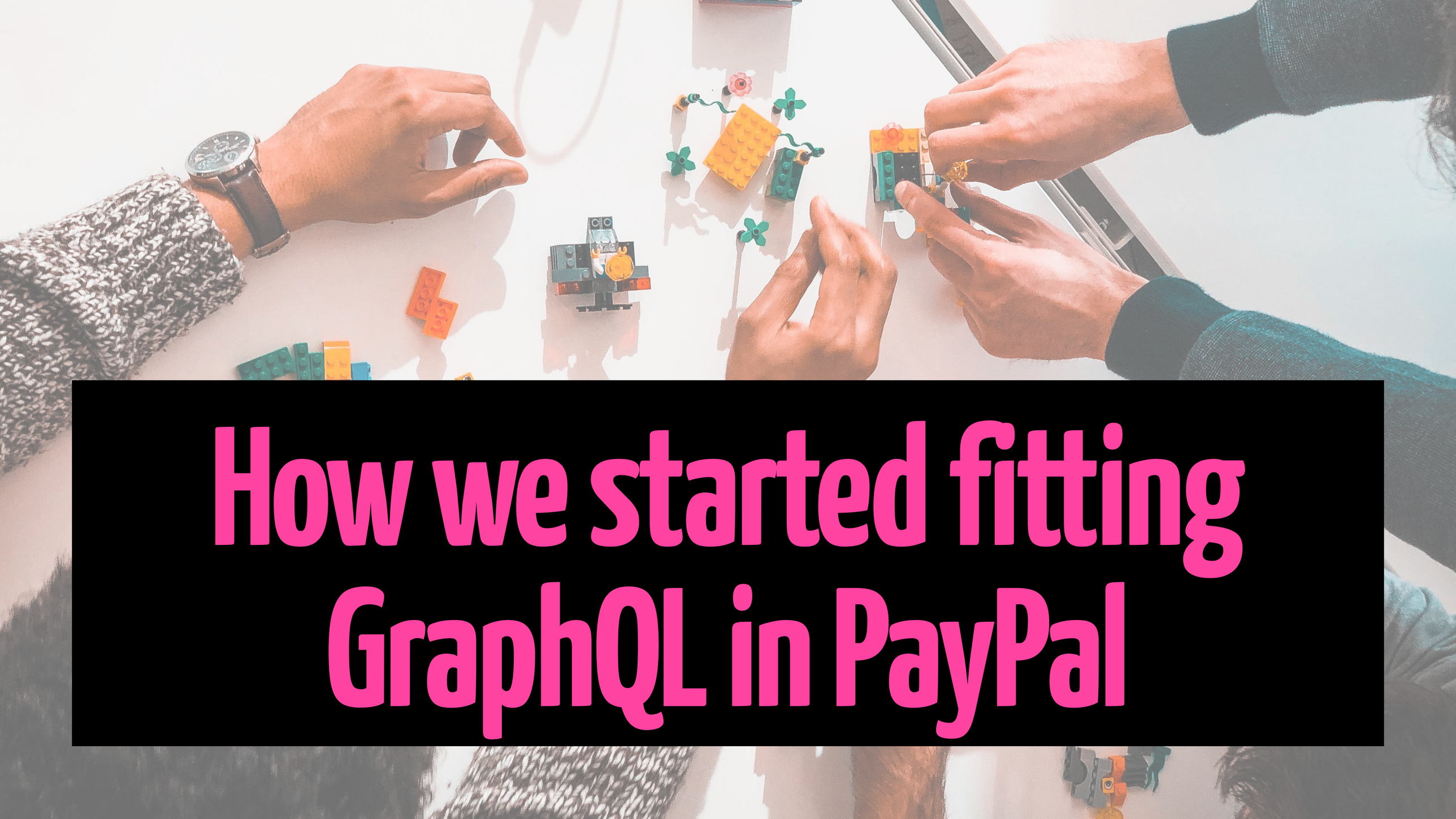
REST API

GraphQL API

REST API



@shrutikapoor08

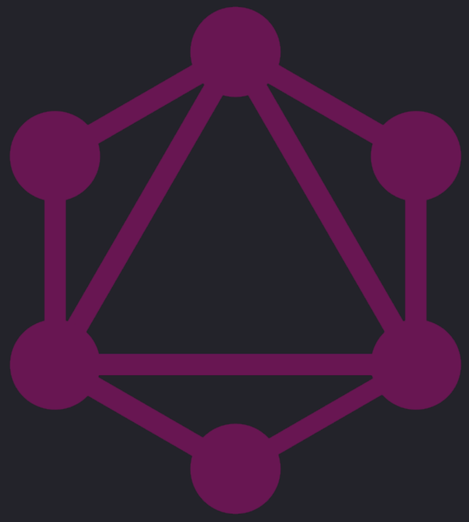


How we started fitting GraphQL in PayPal

Possible options of plugging in GraphQL

Option A: Start converting existing APIs

Option B: Build standalone APIs



Possible options of plugging in GraphQL

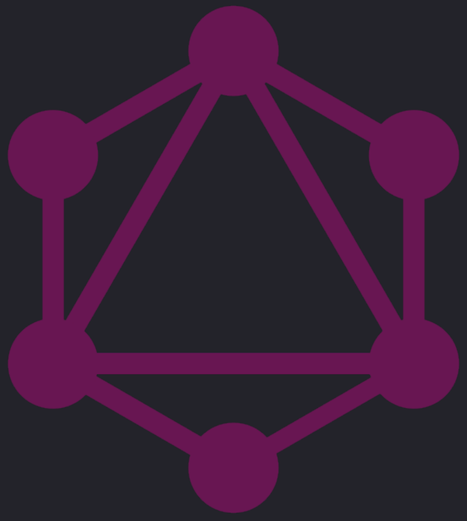
? How many clients are using the API?

? How many downstream APIs are we calling?

? What is the business impact?

? Who are the stakeholders?

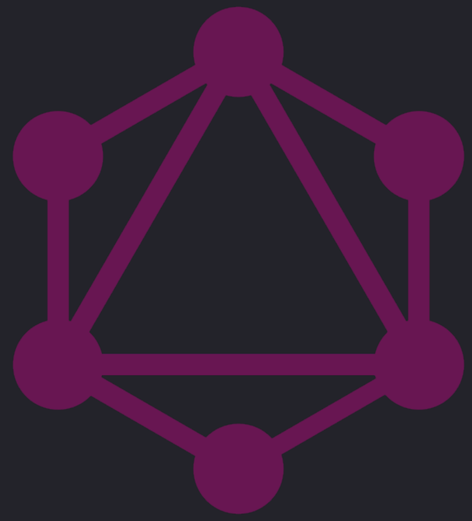
? What is the timeframe?



“First app”

New internal tool

Goal - Make it fast

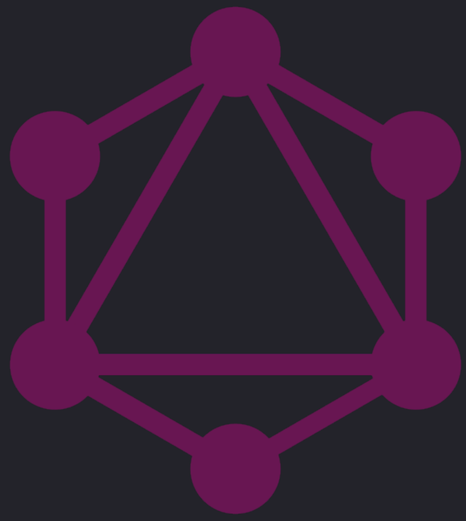


“First app”

New internal tool

Goal - Make it fast

- Less data - no extraneous data
- Less code - no extraneous libraries

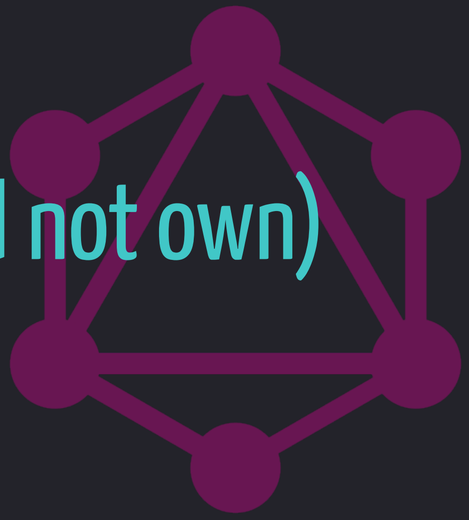


“First app”

Business Logic: We owned the backend REST API.

Authentication: Existing REST API.

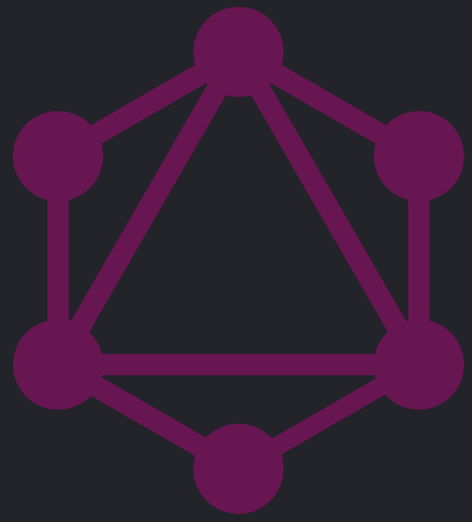
Other enterprise business logic: Downstream REST APIs (did not own)



“First app”

We found that GraphQL shines as an orchestration layer

So that’s what we did!



REST API

REST API

REST API

REST API

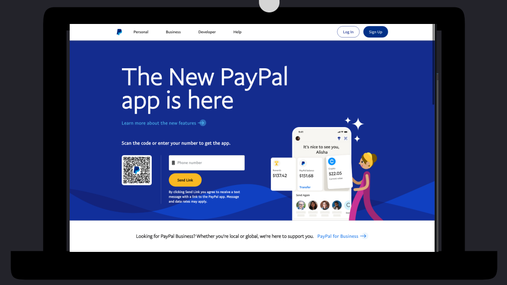
REST API

(Business logic)

Authentication

Analytics

GraphQL API



@shrutikapoor08

“First app”

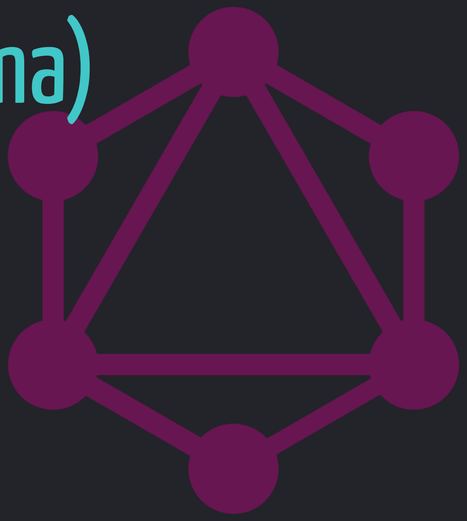
Wrap REST api with a GraphQL API

Client sees and reaps the benefit of GraphQL

Backend developers don’t need to change the way they implement

Frontend and backend developers have a contract (Schema)

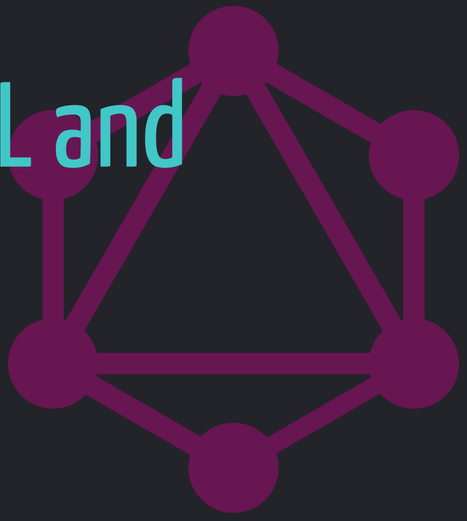
Everyone talks to everybody! Team collaboration 🎉❤️



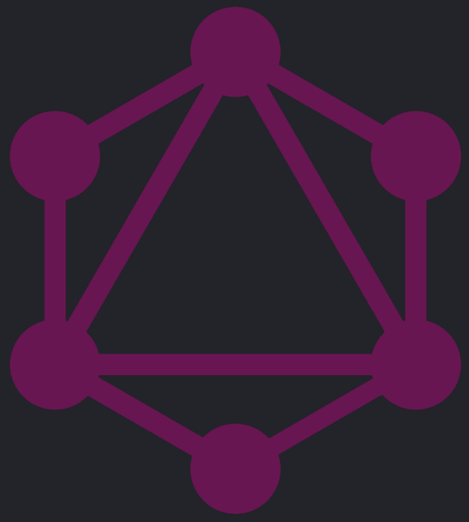
“First app”

We created GraphQL interface to orchestrate that new REST API

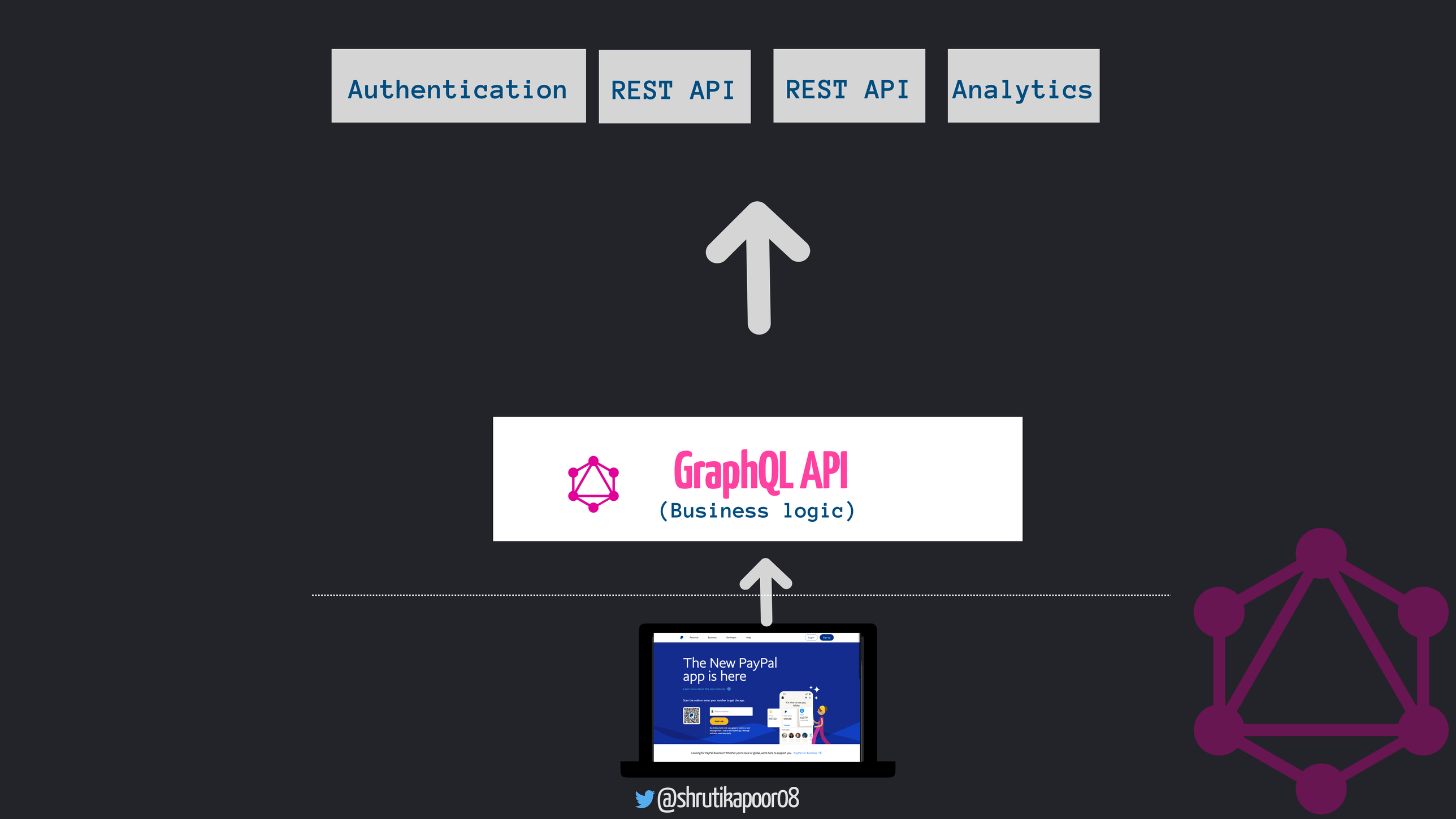
- only deliver data we needed,
- make it easier to evolve API over time
- Make it easy for us to incrementally adopt GraphQL and orchestrate over existing APIs



“Second approach app”



Authentication | REST API | REST API | Analytics



GraphQL API

(Business logic)

@shrutikapoor08

Current Status

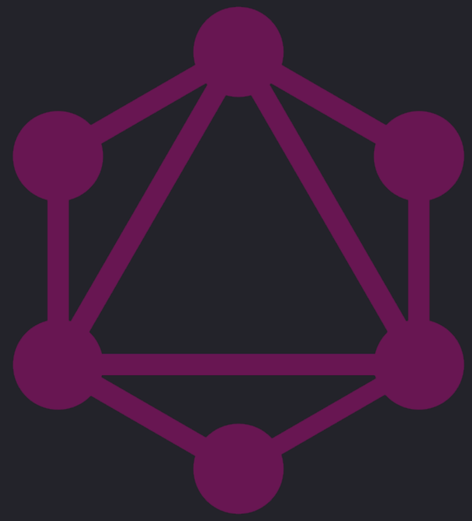
The front tier applications default to GraphQL.

Mid-tier apps are starting to move to GraphQL.

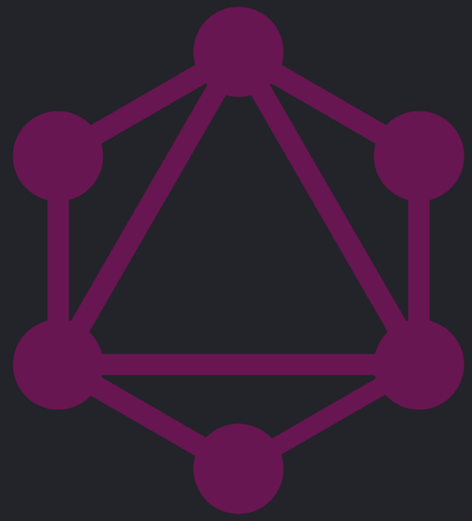
Back end teams still hesitant.

Most teams have their own graphql endpoints.

Plan to move towards a federated graph

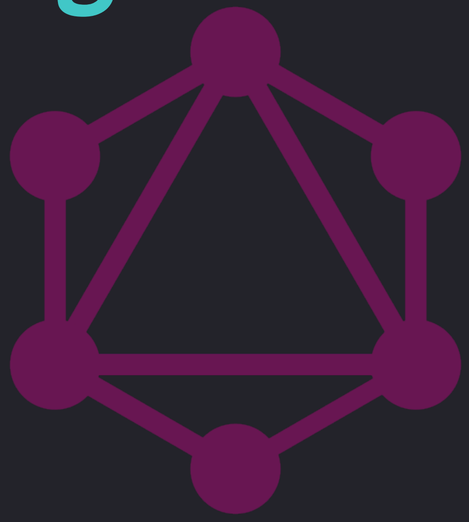


Performance

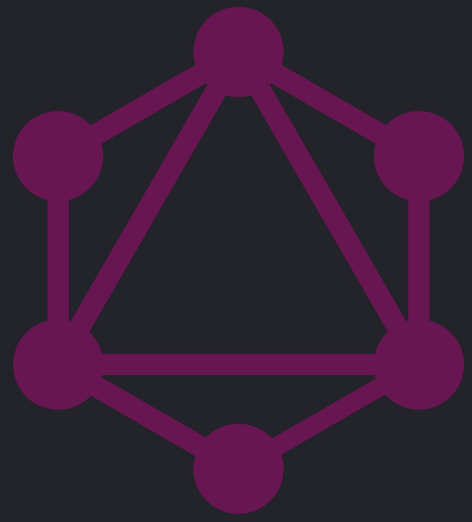


Clients determine size of data = less data

Clients determine shape of data = less logic



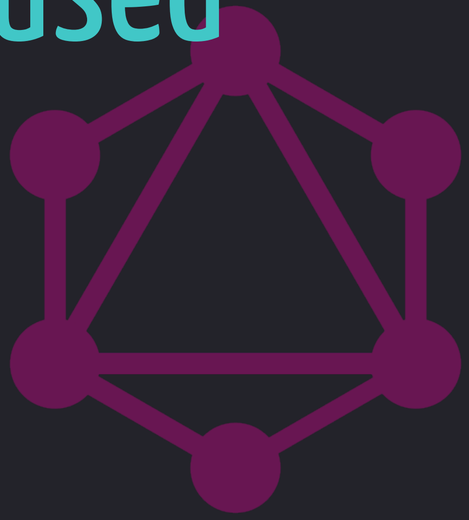
Dev Productivity



GraphiQL: see docs and fire requests

The explorability of docs was awesome!

Evolution of APIs: know what fields are used

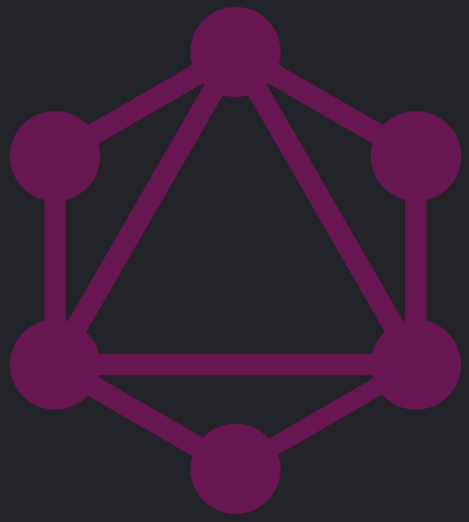


Evolve over time

Established a schema contract

UI drove the schema

GraphQL + REST APIs built in parallel



Lessons learnt

Pick a “first app”

Orchestrate!

Don’t do a 1:1 mapping of Rest <=> GraphQL

Build Schema together

Use built-in GraphQL features = free type safety

CAUTION

GraphQL API will be only as fast as slowest endpoint

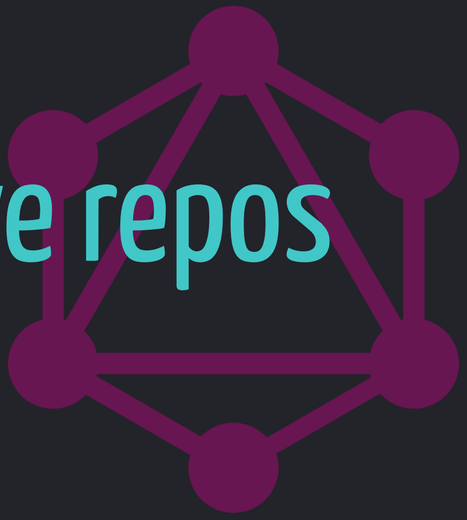
You will have to think about schema in advance.

Challenges of GraphQL

Caching, Error Mapping

You need to know the schema upfront - harder for public APIs

Making One graph when multiple teams have repos



NEXT STEPS

Assess if GraphQL is right for your architecture

Assess if GraphQL is right for your architecture

What are your challenges?

Who is the consumer?

Who has ownership of the APIs?

Where will you put business logic?

Assess if GraphQL is right for your architecture

What company specific apis do you have?

- Eg: authorization, analytics, monitoring, experimentation

Will you save any data by converting to GraphQL?

Do you depend heavily on caching, error mapping and handling?

Assess if GraphQL is right for your company

How will you onboard new teams?

Who is going to enforce standards?

How would you deal with changing schema?

Getting Ready

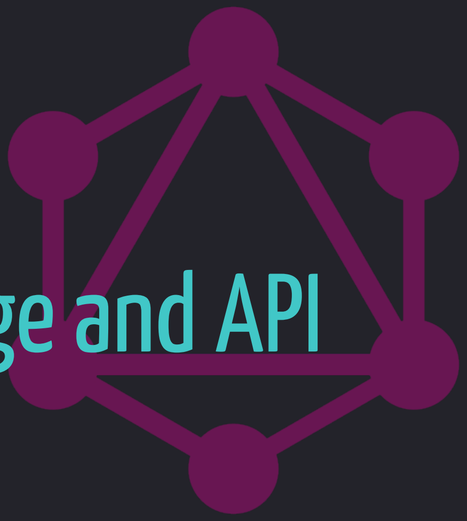
Getting Ready

Do a dogfood exercise with your company. Determine your challenges, “first apps”.

Set a standards team: A team to help with schema stitching, resolving schema conflicts, knowledge transfer

Explore what open source libraries you can use.

Determine how you are going to scale - adoption, knowledge and API



How to sell to your team

Dev productivity

Incremental adoption

Collaboration between teams

Focus on Tooling

https://graphql.org/code/#generic-tools

No versions

Similar to JSON ->
Easy to understand

Get architects and domain experts onboard

FAQs

Myths busted

You need a Graph type database for GraphQL

@shrutikapoor08

GraphQL = SQL?

GraphQL is only for JS developers

@shrutikapoor08

But all my services are in REST

I have sensitive data

Can I limit what APIs are exposed?

I heard GraphQL doesn't support auth

@shrutikapoor08

I don’t want to expose my entire database

To wrap up

No technology is perfect

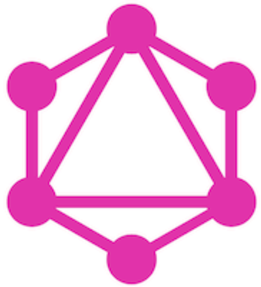
GraphQL is here to stay

Get involved in the community

GraphQL: A success story for PayPal Checkout

Mark Stuart Following

Oct 16, 2018 · 7 min read



From graphql.org

At PayPal, we recently introduced GraphQL to our technology stack.

If you haven't heard of GraphQL, it's a wildly popular alternative to REST APIs that is currently taking the developer world by storm!

At PayPal, GraphQL has been a complete game

PayPal Technology Blog

https://medium.com/paypal-tech

Success stories of companies

- https://medium.com/expedia-group-tech/graphql-component-architecture-principles-homeaway-ede8a58d6fde
- https://medium.com/paypal-engineering/scaling-graphql-at-paypal-b5b5ac098810
- Marc Andre's book: https://book.productionreadygraphql.com/

@shrutikapoor08

twitch.tv/shrutikapoor

JS tinyletter.com/shrutikapoor

Slides at https://github.com/shrutikapoor08/talks

What is(n't) GraphQL - the misconceptions.

With GraphQL, you get exactly what you asked for. It is great for getting multiple resources in one request. In this issue of JSByte, I will address some common misconceptions about GraphQL.

Myth 1: GraphQL only works with graph like structures.

Reality: GraphQL can be used to query a graph database, but it is not it's only use case. The "graph" in GraphQL is used to represent the graph-like structure of data.

Myth 2: GraphQL only works with databases or data sources that are graph based.

Reality: GraphQL can be a wrapper around any data source, including databases. GraphQL is a `query language for your API` - which means it is a syntax of how to ask for data.

bit.ly/shrutinewsletter

@shrutikapoor08

Shruti Kapoor @shrutikapoor08 · Feb 4

Question: Why did the database administrator leave his wife?

She had one-to-many relationships.

#DevJoke

Thank You

@shrutikapoor08