

# GraphQL

eine Alternative zu REST

Stephan Strehler

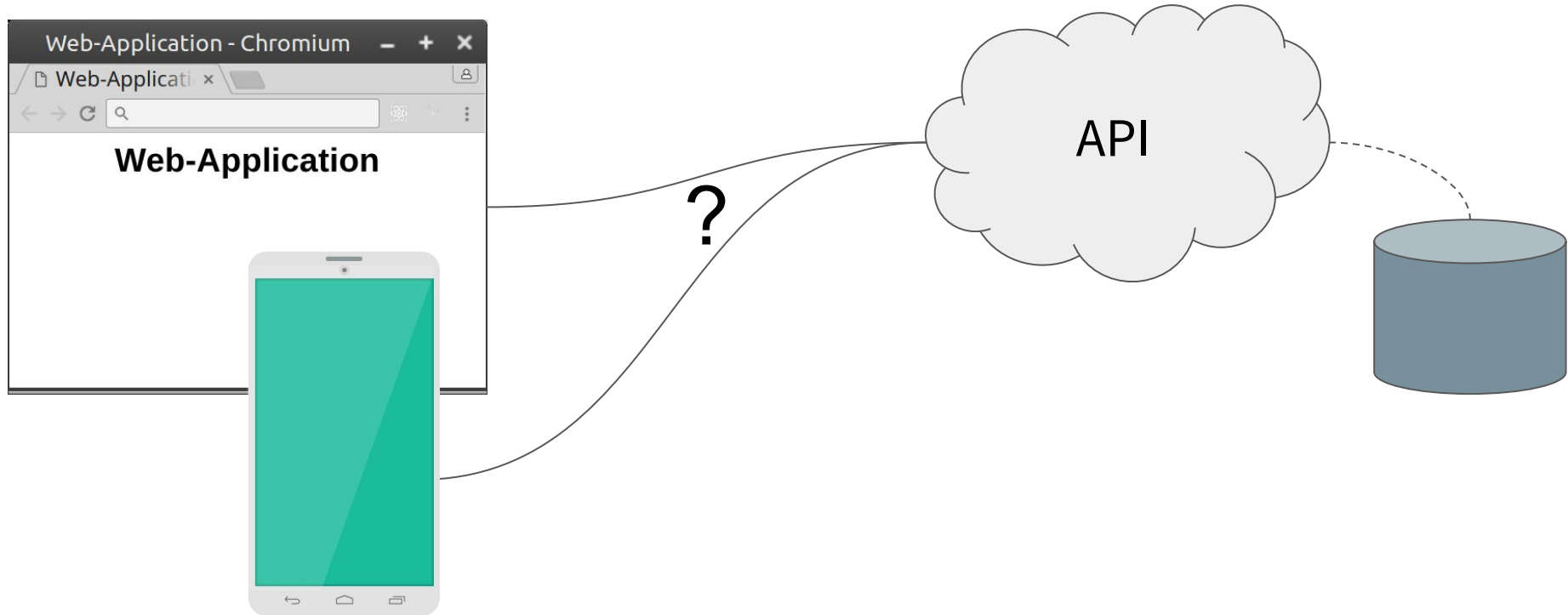
 @StephanStrehler

JUG  
Görlitz 



**Saxonia** Systems

# Themengebiet



# Tagespunkte

1. Vorstellung Webshop
2. Die Lösung mit REST
3. Probleme des Webshops
4. GraphQL
5. Die Lösung mit GraphQL

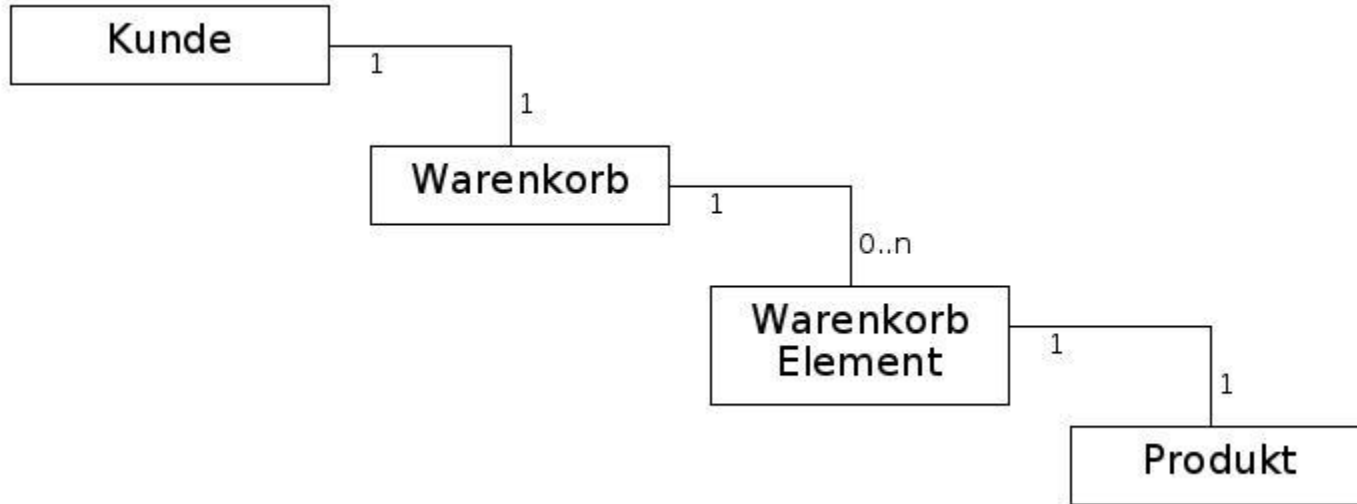
# Beispiel Shopsystem – Einstieg

## Use Cases

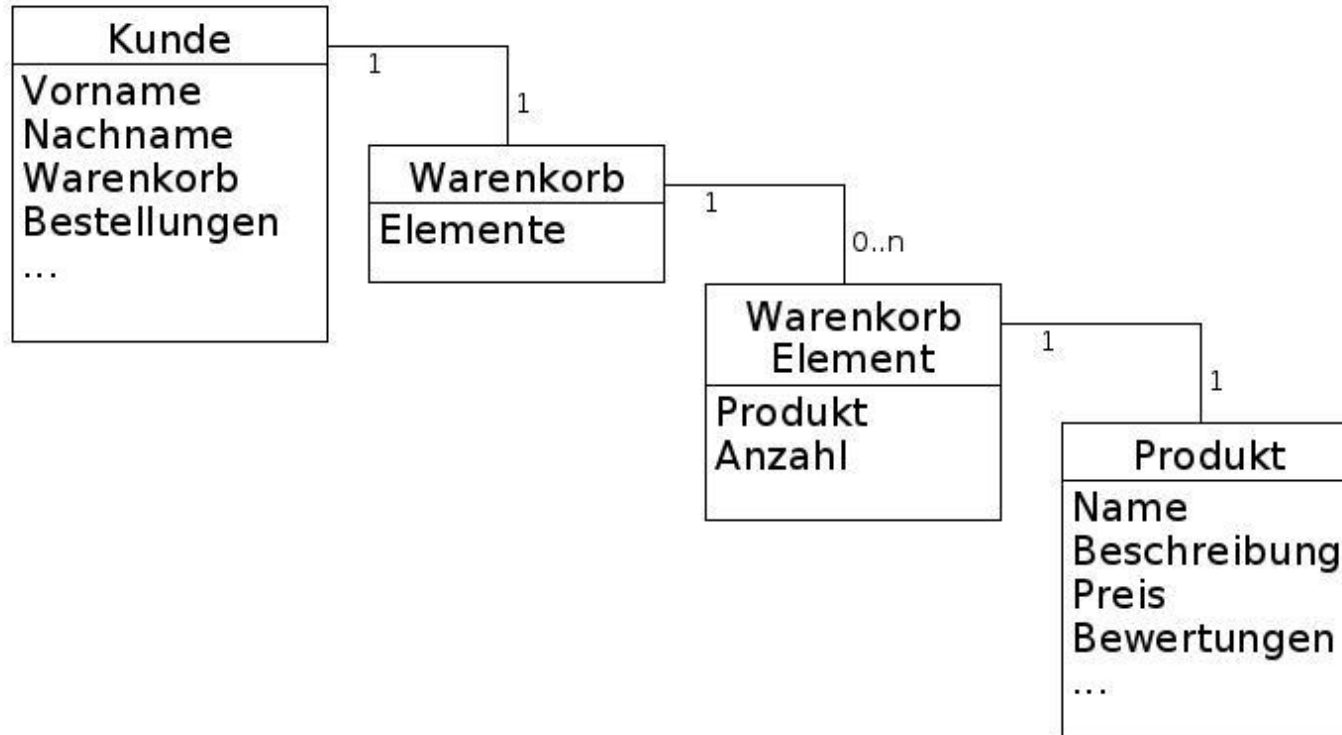
- Kategorien / Produkte anzeigen
- Produkte in Warenkorb legen
- Warenkorb anzeigen
- ...

Link zum Shopsystem: <https://github.com/StevieSteven/graphql-example>

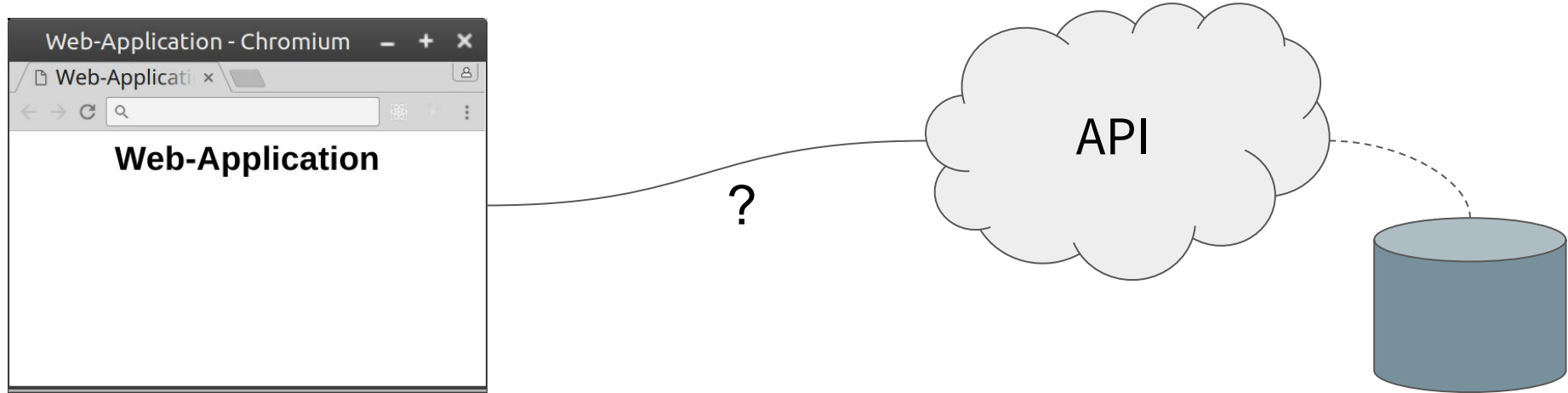
# Shopsystem – Warenkorb



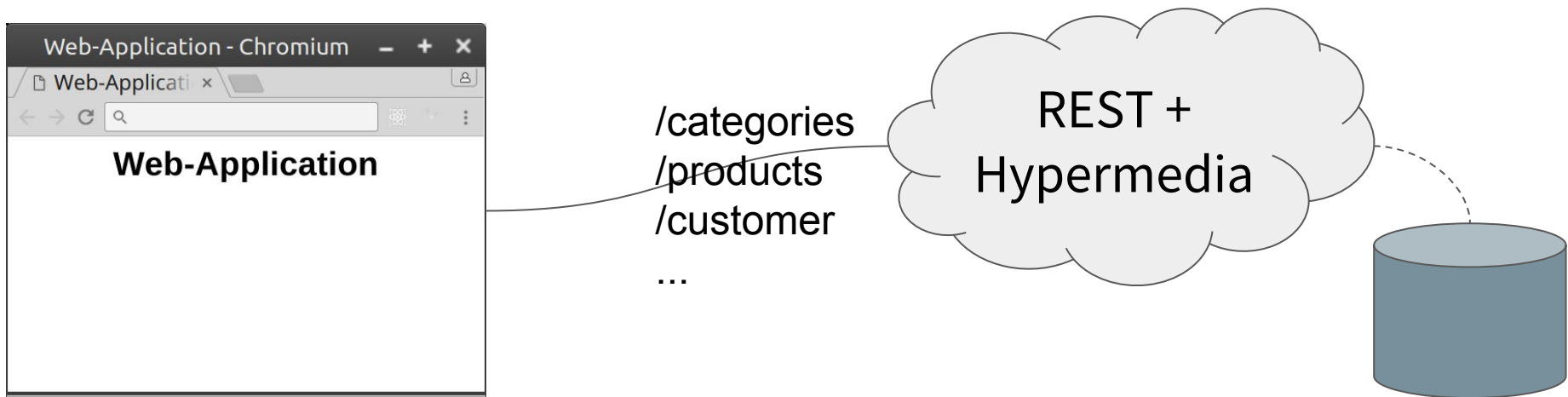
# Shopsystem – Warenkorb



# Shopsystem



# Shopsystem – REST + Hypermedia Entwurf





# Hypermedia?

Request:

/customer/:id

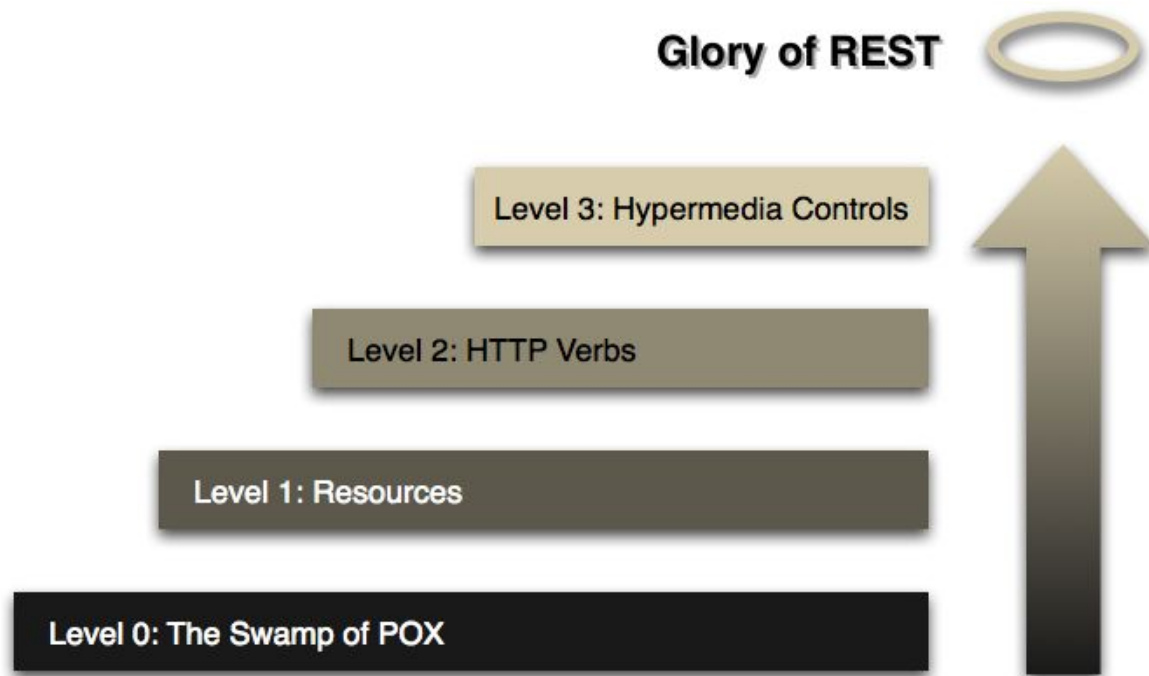
Response:

...

```
_links:{  
  self: {href: "/customer/:id"}  
  cart: {href: "/customer/:id/cart"}  
  ...  
}
```

...

# Richardson Maturity Model



Martin Fowler: <https://martinfowler.com/articles/richardsonMaturityModel.html>

# Vorteile von REST

- Etablierte Technologie
- HTTP als Protokoll
- Caching der Daten auf Netzwerkebene
- Klare Struktur der Schnittstelle
- Lose Kopplung zwischen Client und Server

# Shopsystem –Laden des Warenkorbs



Mein Warenkorb 

Request:

/cart

Response:

n Links zu /cart\_item/:id

# Shopsystem –Laden des Warenkorbs



## Mein Warenkorb (5 Produkte)

Rustic Concrete Chicken

Preis: 726 € pro Stück

Beschreibung: Qui voluptatum nulla non consequatur necessitatibus aliquam eum quibusdam aperiores. Ipsa enim quia minus esse nuncquam sequi tempora commodi ut. Velit voluptatem doloribus qui. Aut facilis voluptatum. Quasi labore voluptas. Sit debita necessitatibus quia ut dolorum accusantium dolorum.

1 Stück für

726 €

Licensed Granite Chair

Preis: 313 € pro Stück

Beschreibung: Nostrum voluptas minus voluptas qui. Repellat ipsum ipsa. Aperiam commodi neque voluptatem. Pariatur nostrum consequatur voluptatem sit eum.

2 Stück für

626 €

Practical Soft Pizza

Preis: 218 € pro Stück

Beschreibung: Porro occaecati ea atque quasi delectus consequatur et. Cumque enim sapiente nemo nulla. Culpa dolores nuncquam itaque et corrupti exceptat. Eos tenetur ea nostrum dignissimos qui dolorum velit et consequatur. Quosdam et non.

4 Stück für

872 €

Handmade Fresh Car

Preis: 79 € pro Stück

Beschreibung: Sit ad fugiat commodi in atque. Aut et voluptatem voluptas molestiae reiciendis. Iste rerum cum iure reprehenderit explicabo ea quis esse. Voluptate impedit enim accusamus fugit.

4 Stück für

316 €

Request:

`n /cart_items/:id`

Response:

- `n cart_items`

- je 1 Link zu `/product/:id`

# Shopsystem –Laden des Warenkorbs



## Mein Warenkorb (5 Produkte)

Rustic Concrete Chicken

Preis: 726 € pro Stück

**Beschreibung:** Qui voluptatum nulla non consequatur necessitatibus aliquam eum quibusdam asperiores. Ipsa enim quia minus esse numquam sequi tempora commodi ut. Velit voluptatem doloribus qui. Aut facilis voluptatum. Quasi labore voluptas. Sit debitis necessitatibus quia ut dolorum accusantium dolorem.

1 Stück für

726 €

Licensed Granite Chair

Preis: 313 € pro Stück

**Beschreibung:** Nostrum voluptas minus voluptas qui. Repellat ipsam ipsa. Aperiam commodi neque voluptatem pariatur nostrum consequatur voluptatem sit eum.

2 Stück für

626 €

Practical Soft Pizza

Preis: 218 € pro Stück

**Beschreibung:** Porro occaecati ea atque quasi delectus consequatur et. Cumque enim sapiente nemo nulla. Culpa dolores numquam itaque et corrupti excepturi. Eos tenetur ea nostrum dignissimos qui dolorum velit et consequatur. Quaerat et non.

4 Stück für

872 €

Handmade Fresh Car

Preis: 79 € pro Stück

**Beschreibung:** Sit ad fugiat commodi in atque. Aut et voluptatem voluptas molestiae reiciendis. Iste rerum cum iure reprehenderit explicabo ea quis esse. Voluptate impedit enim accusamus fugiat.

4 Stück für

316 €

Request:

n /products/:id

Response:

- n products

# REST + Hypermedia: Problem 1

$1 + (2 * n)$  Requests  
notwendig um Warenkorb  
anzuzeigen!

n: Anzahl an Produkten

# REST + Hypermedia: Problem 2

Zu viele Informationen!

Viele Daten...

Links zu weiteren Ressourcen...

... werden nicht benötigt!



# REST + Hypermedia: Problem 3

View spezifische Ressourcen

/search

/category\_list

...

⇒ Kopplung zwischen Server und Client

# REST + Hypermedia: Problem 4

Sortierung der Produkte:

/products?orderBy=???&???=???

⇒ Parameter orderBy unbekannt

# REST + Hypermedia

Wir lösen die Probleme!

# REST + ~~Hypermedia~~: Lösung Problem 1

Wir nutzen REST...

... aber wir liefern mit dem  
Request "cart" alle Informationen  
mit!

# REST + ~~Hypermedia~~: Lösung Problem 1

Was liefert /cart?

Spätestens bei getrennten  
Entwicklerteams scheitert der  
Versuch.

Es werden noch mehr unnötige  
Daten geliefert.

GESCHETTERT

# REST + Hypermedia: Lösung Problem 2

Zu viele Informationen?

Lösung: View spezifische Ressourcen

# REST + Hypermedia: Lösung Problem 3 + 4

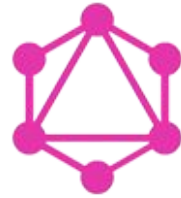


# REST + Hypermedia: Zusammenfassung

1. Viele Requests notwendig
2. Zu viele Daten werden übermittelt
3. View-spezifische Ressourcen
4. Keine Typisierung



neuer Ansatz



GraphQL

# Vorstellung

- Abfragesprache für (Web-)APIs
- Entwickelt von Facebook
- Seit 2012 bei Facebook Mobile App im Einsatz
- Seit Ende 2015 OpenSource

# Vorstellung

- Spezifikation existiert (letzte Version: Okt. 2016)
- graphql.js als offizielle Implementationsprache von FB
- Bibliotheken auch in anderen Sprachen vorhanden

# Shopsystem: Frameworks



# Als Verbesserung

## Wie löst GraphQL die Probleme?

# Generelles

- Besitzt ein einzigen Einstiegspunkt
- GET oder POST als HTTP Methode
- Trennung von Query und Mutation

# Technik

Schema

Resolver

Query

Definiert serverseitige Schnittstelle:

```
type Category {  
  uuid: ID!,  
  name: String!,  
  numberOfProducts: Int!,  
  products: [Product!]  
}
```

```
category(uuid: ID!): Category
```

# Technik

Schema

Resolver

Query

Funktion zum Bereitstellen der Daten:

```
category(_, {uuid}) {  
    return Category.findByUUID(uuid);  
},
```



# Technik

Schema

Resolver

Query

Clientseitige Abfrage:

```
{  
  category(uuid:"...") {  
    name  
    numberOfProducts  
    products {  
      name  
      price  
    }  
  }  
}
```

# GraphQL

## Serverseitig

- Typen
- Schemaerzeugung
- Resolver
- Ablauf Request

# Typen

## Scalare Typen

Int, Boolean, String, Float

⇒ keine extra Resolve  
Funktion nötig

## komplexe Typen, Enums

⇒ Resolve Funktion  
notwendig

# Schema Programmatisch (graphql-java)

```
GraphQLObjectType categoryType = GraphQLObjectType.newObject()  
    .name("Category")  
    .field(GraphQLFieldDefinition.newFieldDefinition()  
        .name("uuid")  
        .type(GraphQLID))  
    .field(GraphQLFieldDefinition.newFieldDefinition()  
        .name("name")  
        .type(GraphQLString))  
    .field(GraphQLFieldDefinition.newFieldDefinition()  
        .name("products")  
        .type(new GraphQLList(  
            new GraphQLTypeReference("Product")  
        ))  
    .build();
```

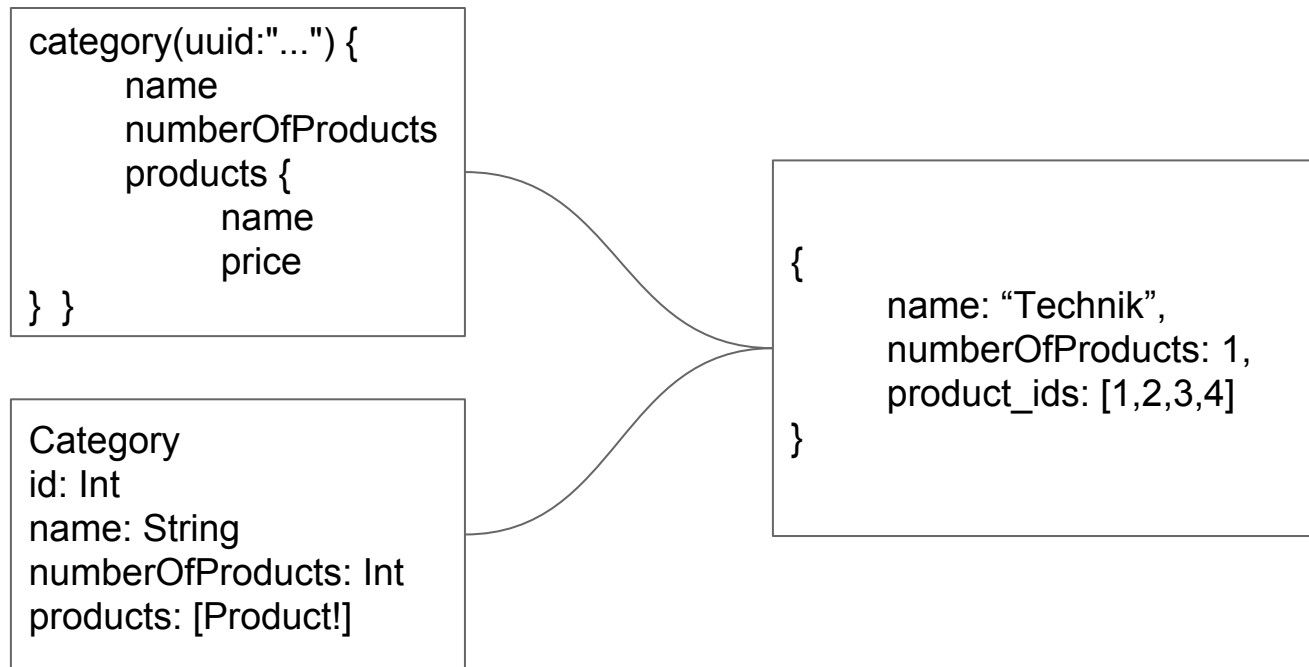
# Schema Programmatisch (graphql.js)

```
const CategoryType = new GraphQLObjectType({  
  name: 'Category',  
  fields: {  
    uuid: { type: GraphQLID },  
    name: { type: GraphQLString },  
    products: { type: new GraphQLList(ProductType) }  
  }  
})
```

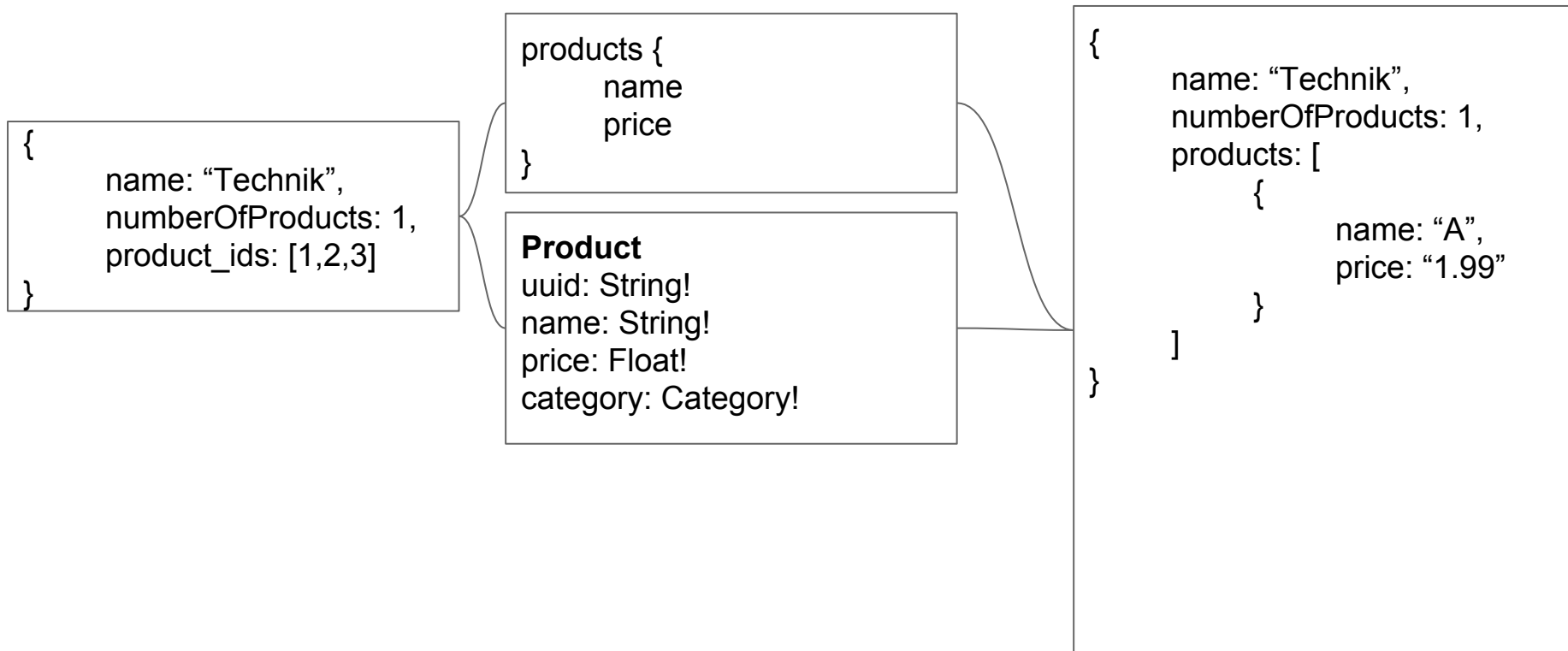
# Resolver (Mit Apollo GraphQL)

```
const resolveFunctions = {  
  Query: {  
    category(_, {uuid}) {  
      return Category.findByUUID(uuid);  
    }  
  },  
  Category: {  
    products: {  
      resolve(root) {  
        return Category.products(root);  
      }  
    }  
  }  
}
```

# Ablauf Request

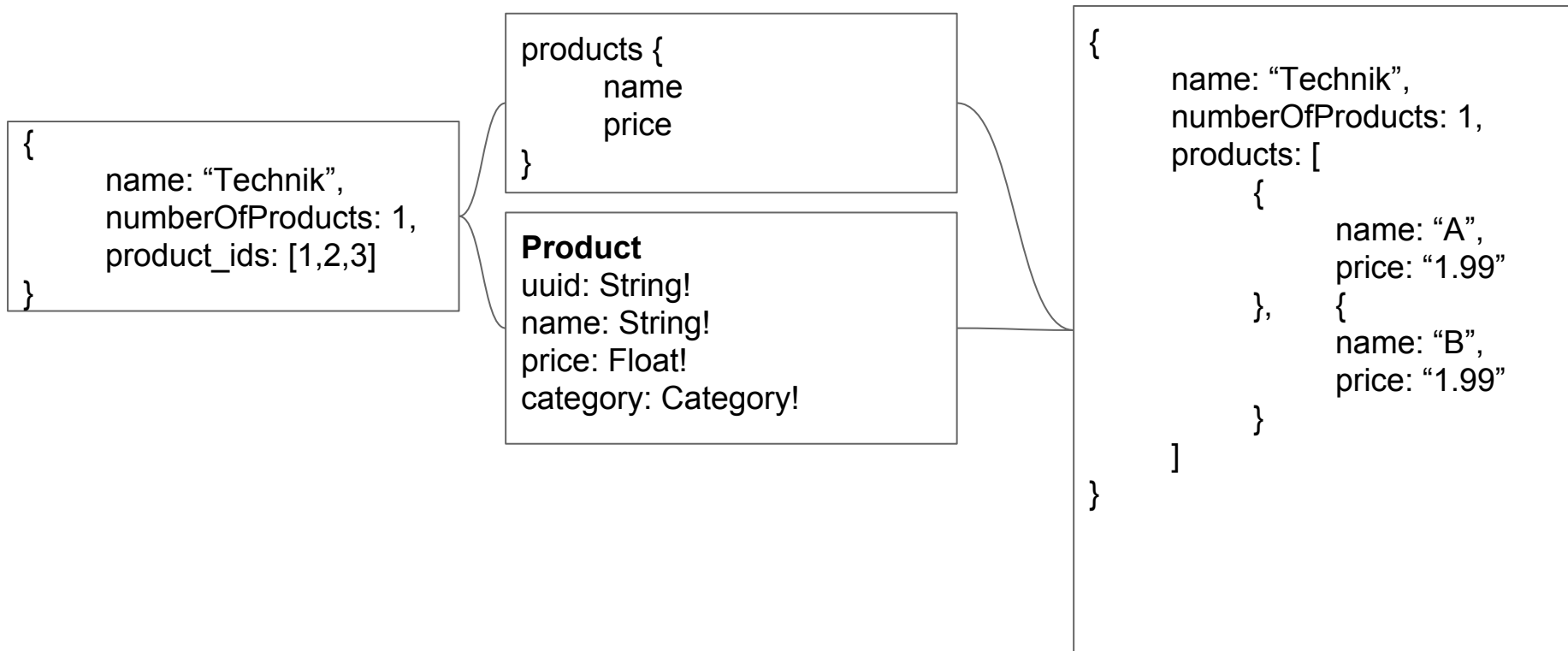


# Ablauf Request

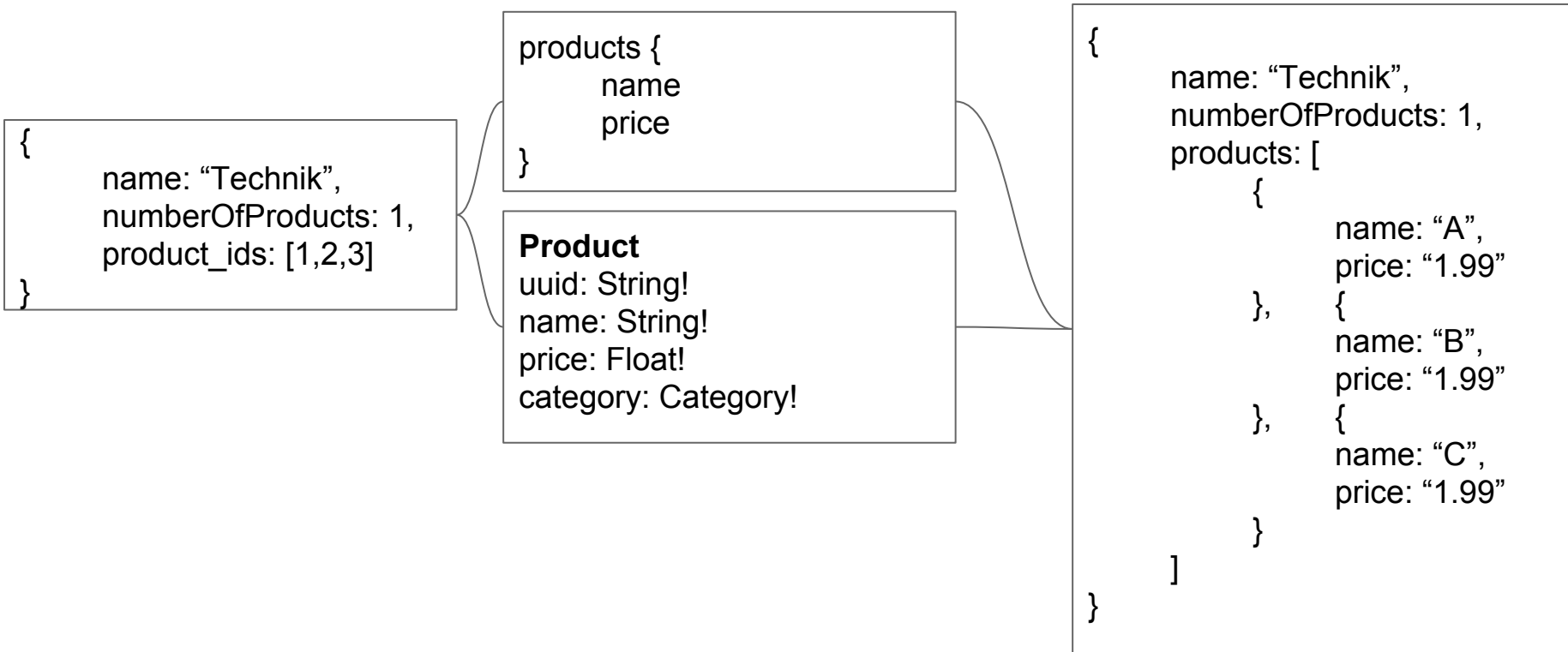




# Ablauf Request



# Ablauf Request



# GraphQL

Clientseitig

- Query
- Query Optimierung
- Mutations

# Query

- Jede Komponente besitzt eigenen Query
- Query fragt genau die benötigten Informationen ab

# Query Optimierung

[Suche](#)[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

## Mein Warenkorb (5 Produkte)

Small Fresh Soap

**Preis:** 552 € pro Stück

**Beschreibung:** Excepturi aperiam quibusdam nesciunt non. Accusantium hic quisquam possimus veritatis animi atque omnis. Modi itaque officia aut ratione. Non non earum dolor beatae mollitia sint voluptatem aut, Ipsam voluptas soluta rerum vero corporis sed temporibus eveniet ea. Voluptas aut alias.

2 Stück für  
1104 €

Gorgeous Rubber Chicken

**Preis:** 189 € pro Stück

4 Stück für  
756 €

# Query Optimierung

## Navigationsbereich:

```
shoppingcard {  
  items {  
    quantity  
  }  
}
```

## Warenkorb:

```
shoppingcard {  
  items {  
    quantity  
    product {  
      uuid  
      name  
      description  
      price  
    }  
  }  
}
```

# Query Optimierung

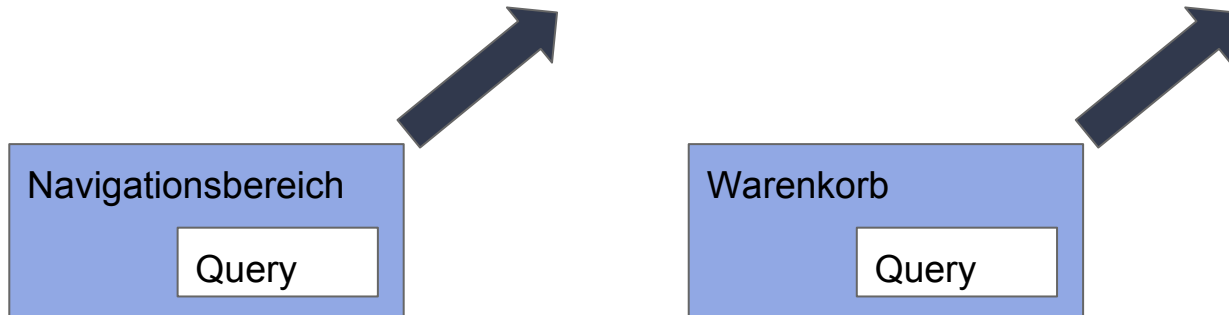
## Navigationsbereich:

```
shoppingcard {  
  items {  
    quantity  
  }  
}
```

## Warenkorb:

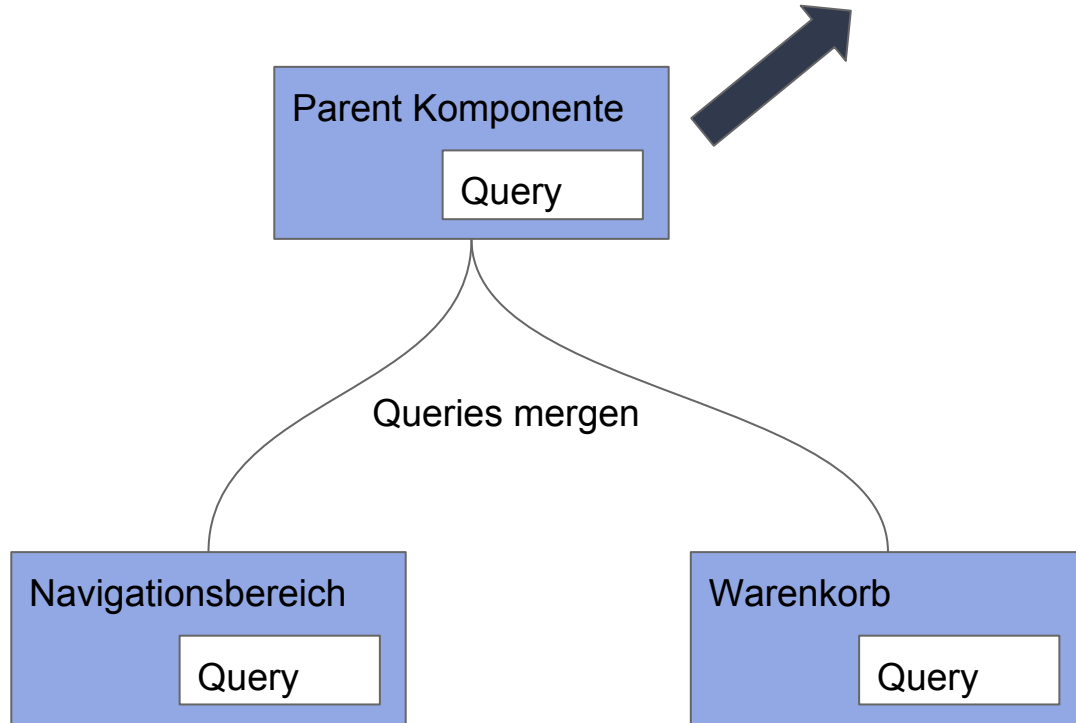
```
shoppingcard {  
  items {  
    quantity  
    product {  
      uuid  
      name  
      description  
      price  
    }  
  }  
}
```

# Query Optimierung

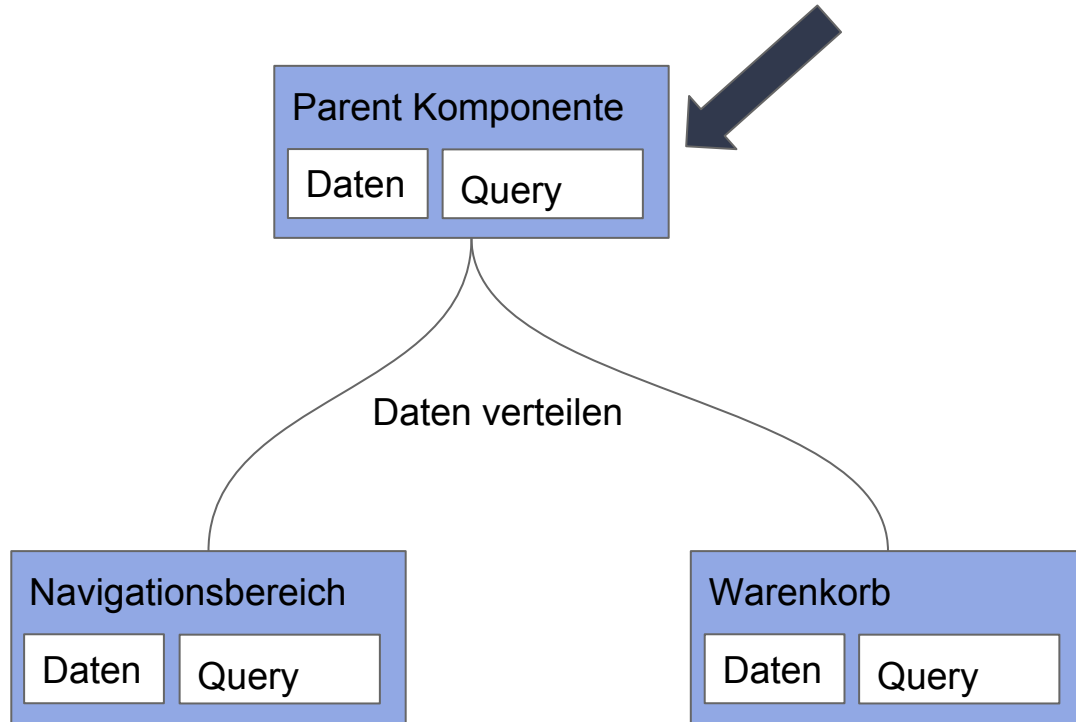




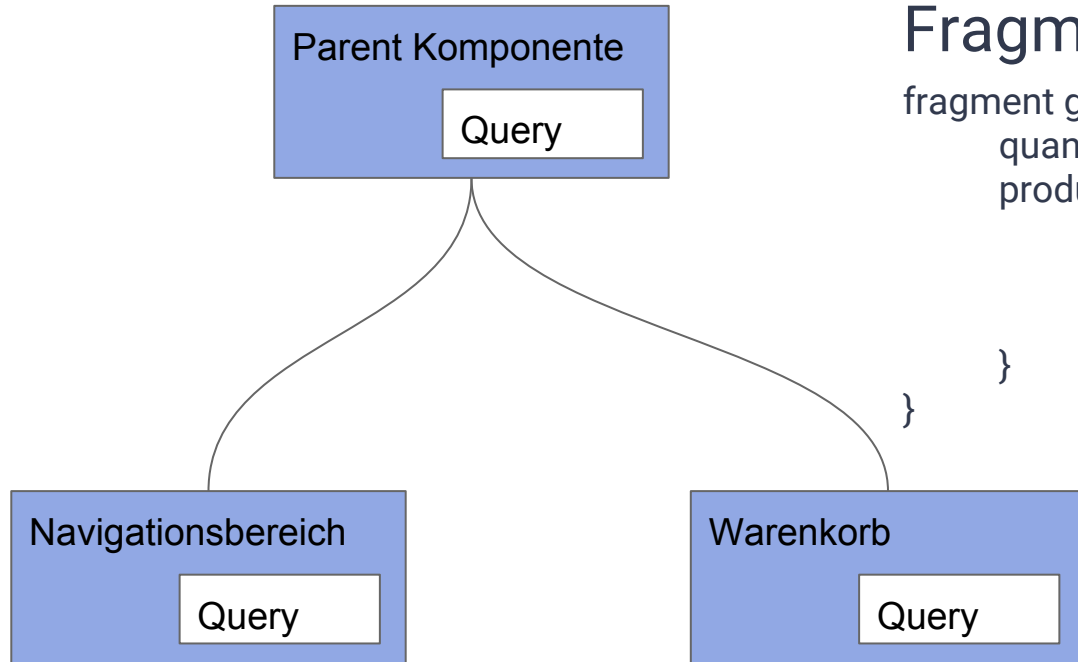
# Query Optimierung



# GraphQL - Query Optimierung



# Query Optimierung



## Fragments:

```
fragment generalFields on ShoppingCartElement {  
  quantity  
  product {  
    uuid  
    price  
    name  
  }  
}
```

# Query Optimierung

Aktueller Stand:

- Funktioniert noch nicht problemlos
- Problem: Wann werden die Daten benötigt?

# Query Optimierung

[Suche](#)[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

## Handcrafted Rubber Computer

[In den Warenkorb legen](#)

### Allgemeine Informationen

**Preis:** 867€

**Beschreibung:** Ea et sit distinctio sed error. Rerum praesentium modi consequatur molestiae quasi possimus ut dignissimos. Nihil nulla et et magnam qui est modi. Error et ab ut est mollitia cupiditate et quam. Aliquid aut praesentium minima sit et quam vel deserunt omnis. Id quia consequatur et ipsam hic illum ut quia magnam.

### Andere Nutzer sagen



von Sammie Greenholt

Minus quibusdam unde quam animi distinctio. Quo omnis ad. Voluptatum nostrum similique corrupti eum commodi error officia et. Recusandae sunt asperiores laboriosam fugit.

# Mutations

- Ähnlich Queries, einziger Unterschied:
- Root Element ist nicht Query, sondern mutation

⇒ Vergleichbar mit dem CQRS Pattern

# Mutations

```
schema {  
  query: Query  
  mutation: Mutation  
}
```

```
type Query {  
  ...  
}
```

```
type Mutation {  
  ...  
}
```

# Mutations

## Abfrage an Server

```
const addProductToCartQuery = gql`  
  mutation AddProductToCart($uuid: ID!, $quantity: Int!) {  
    addProductToCart(uuid: $uuid, quantity: $quantity) {  
      uuid  
    }  
  }  
`;  
;
```



# Mutations

## Aufruf der Mutation

```
const handleCartBtn = () => {  
  this.props.mutate({  
    variables: {  
      uuid: this.props.product.uuid,  
      quantity: this.state.numberOfProducts  
    }  
  }).then(({data}) => {  
    ...  
  }).catch((error) => {  
    ...  
  });  
};
```

# Mutations

[Suche](#)[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

## Handcrafted Rubber Computer

 ▾[In den Warenkorb legen](#)

### Allgemeine Informationen

**Preis:** 867€

**Beschreibung:** Ea et sit distinctio sed error. Rerum praesentium modi consequatur molestiae quasi possimus ut dignissimos. Nihil nulla et et magnam qui est modi. Error et ab ut est mollitia cupiditate et quam. Aliquid aut praesentium minima sit et quam vel deserunt omnis. Id quia consequatur et ipsam hic illum ut quia magnam.


### Andere Nutzer sagen



von Sammie Greenholt

Minus quibusdam unde quam animi distinctio. Quo omnis ad. Voluptatum nostrum similique corrupti eum commodi error officia et. Recusandae sunt asperiores laboriosam fugit.

# Mutations

  [Suche](#)

[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

beeinflusst

[In den Warenkorb legen](#)


## Handcrafted Rubber Computer

### Allgemeine Informationen

**Preis:** 867€

**Beschreibung:** Ea et sit distinctio sed error. Rerum praesentium modi consequatur molestiae quasi possimus ut dignissimos. Nihil nulla et et magnam qui est modi. Error et ab ut est mollitia cupiditate et quam. Aliquid aut praesentium minima sit et quam vel deserunt omnis. Id quia consequatur et ipsam hic illum ut quia magnam.

### Andere Nutzer sagen



von Sammie Greenholt

Minus quibusdam unde quam animi distinctio. Quo omnis ad. Voluptatum nostrum similique corrupti eum commodi error officia et. Recusandae sunt asperiores laboriosam fugit.

# GraphQL

Fazit

# Fazit – Vorteile

- Anzahl der Requests wurden minimiert
- Datenmenge reduziert
- “maßgeschneiderte” Daten
- Saubere Komponententrennung möglich
- exakte Definition der Schnittstelle
- Schnittstellenexplorer (GraphiQL)
- Verbesserte parallele Arbeitsprozesse in der Entwicklung

# Fazit – Nachteile

- Sehr hohe Entwicklungsgeschwindigkeit
- Kinderkrankheiten
- Serverauslastung kann höher sein
- Kein Cache?

# Cache

- Netzwerk Cache nicht möglich
- Cache muss von Client Bibliothek übernommen werden

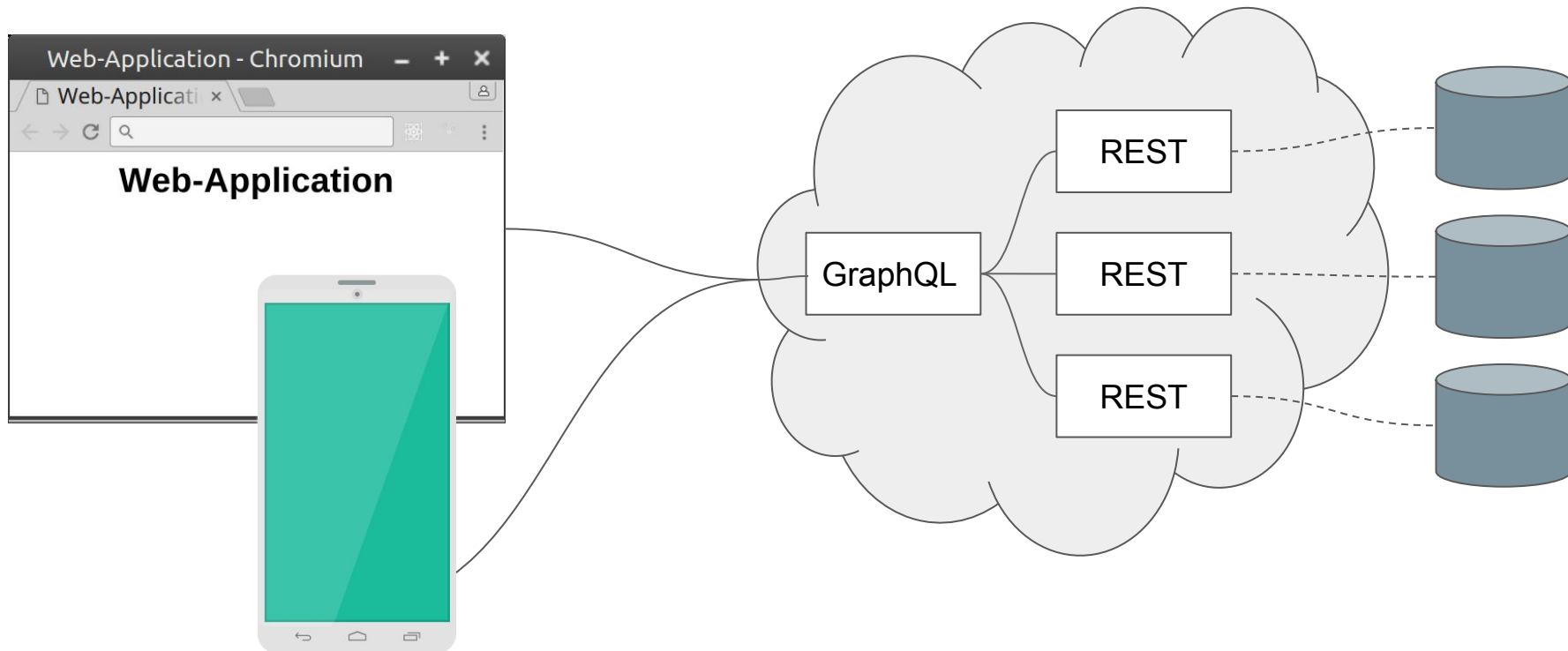
⇒ Komplexität von Bibliotheken steigt

# GraphQL

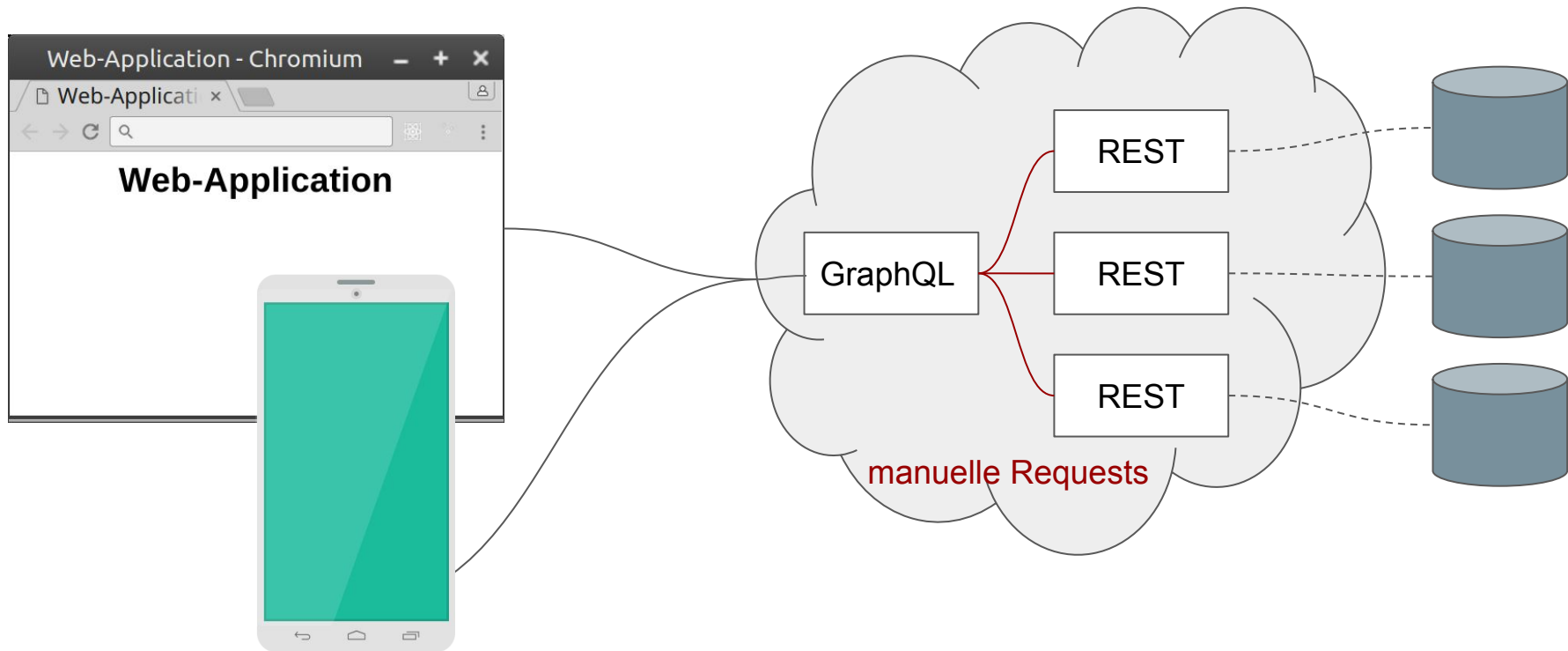
neue Möglichkeiten



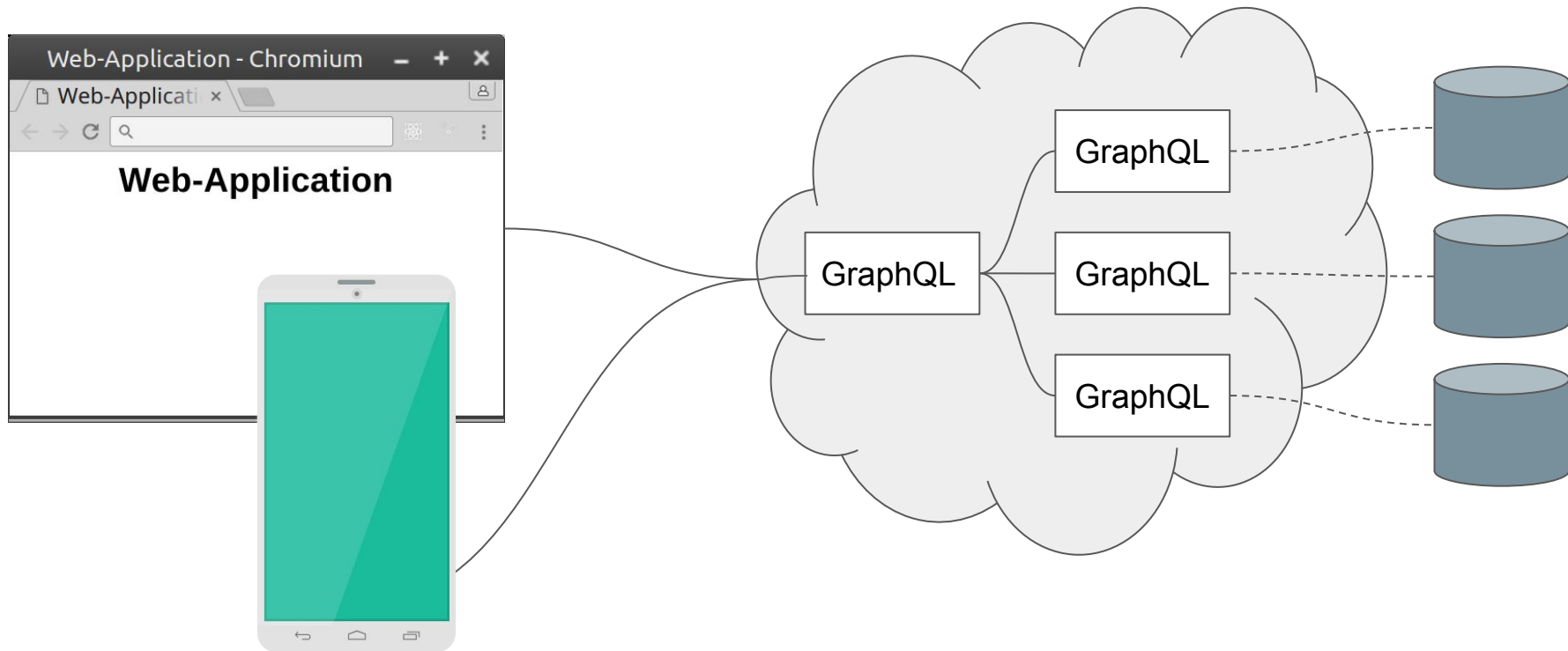
# GraphQL Schema Stitching



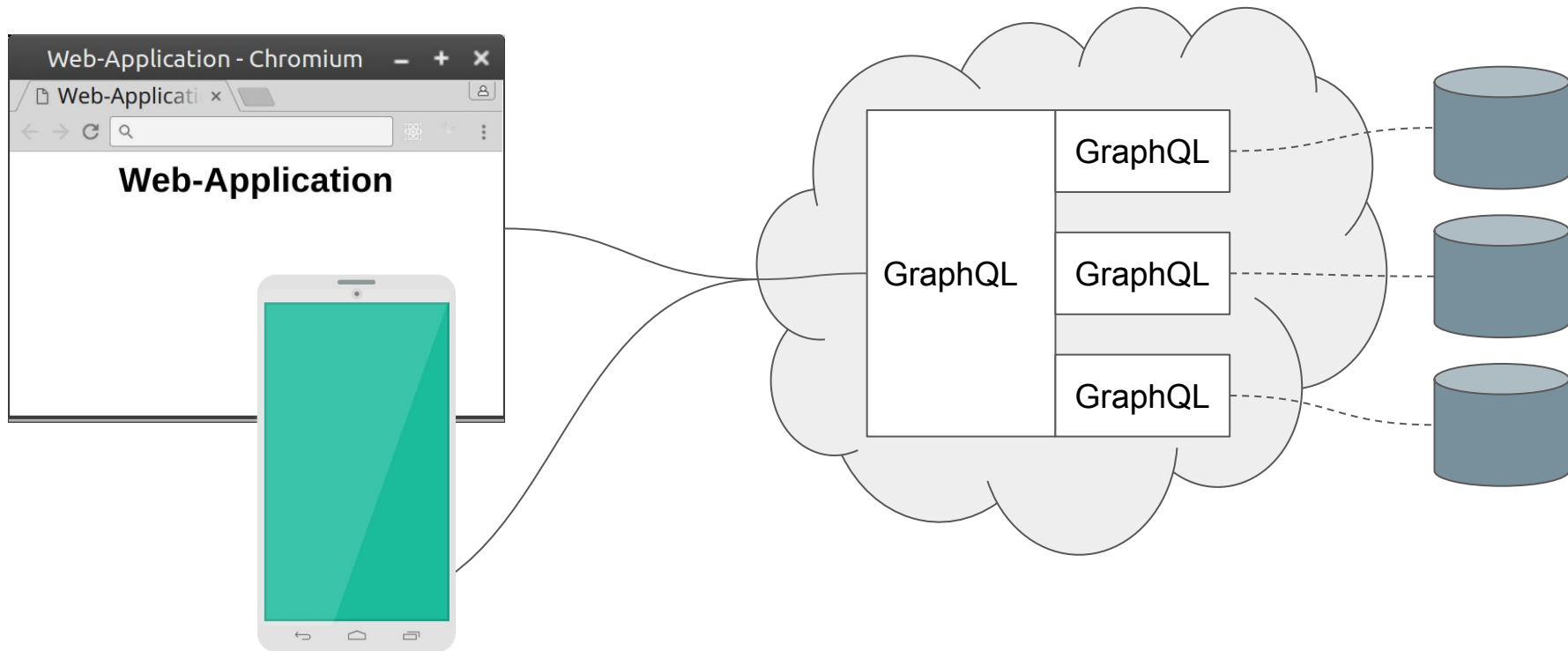
# GraphQL Schema Stitching



# GraphQL Schema Stitching



# GraphQL Schema Stitching



# Schema Stitching

- *createRemoteSchema* Teil von *graphql-tools* (Apollo)
- Laden von allen Schemas anderer Servern
- Koppeln von Schemas
- Requests werden automatisch weitergeleitet

Beispiel: <https://github.com/StevieSteven/graphql-schema-stitching>

# Abschluss

# Abschluss

## Take Aways

- GraphQL besteht aus Schema, Resolver und Query
- GraphQL besitzt statisch definiertes Schema
- Aufbau Query richtet sich nach Schema
- Komplexerer Clientbibliotheken nötig

# Abschluss

## Further Reading

Manuel Mauky, GraphQL: <http://bed-con.org/2017/files/slides/Mauky-GraphQL.pdf>

Apollo GraphQL:

<https://www.apollographql.com/>

React und GraphQL:

<https://dev-blog.apollodata.com/seamless-integration-for-graphql-and-react-6ffc0ad3fead>



# Fragen?

Stephan Strehler  
@StephanStrehler

JUG  
Görlitz



**Saxonia** Systems