

AWS
re:Invent



FWM201

Power modern serverless applications with GraphQL and AWS AppSync

Ed Lima

Senior Product Manager

AWS

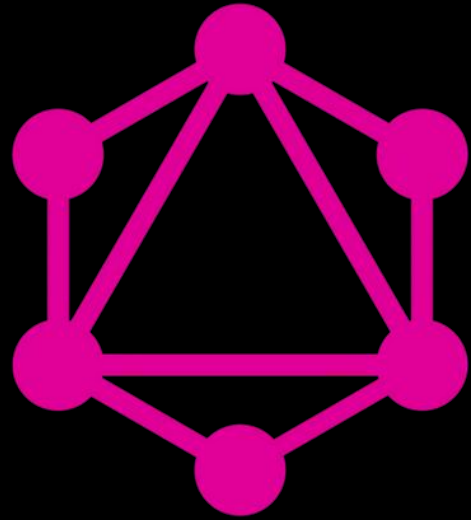
Agenda

GraphQL

AWS AppSync

Features, integrations, and security

Developer tools: Amplify and AWS CDK



GraphQL



GraphQL is a query language for APIs and
a runtime for fulfilling those queries
with your existing data



Describe your data

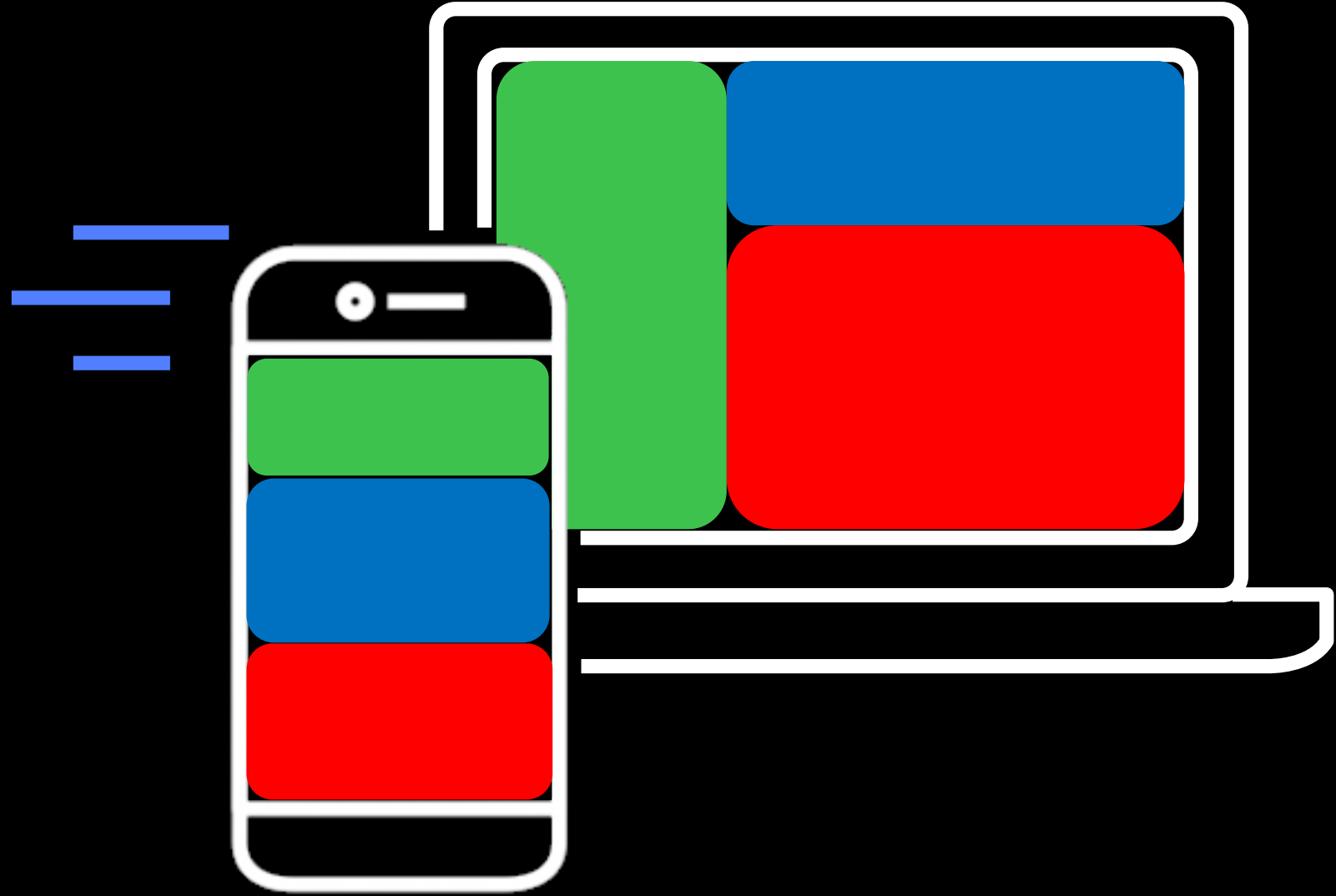
```
type Project {  
  name: String  
  tagline: String  
  contributors: [User]  
}
```

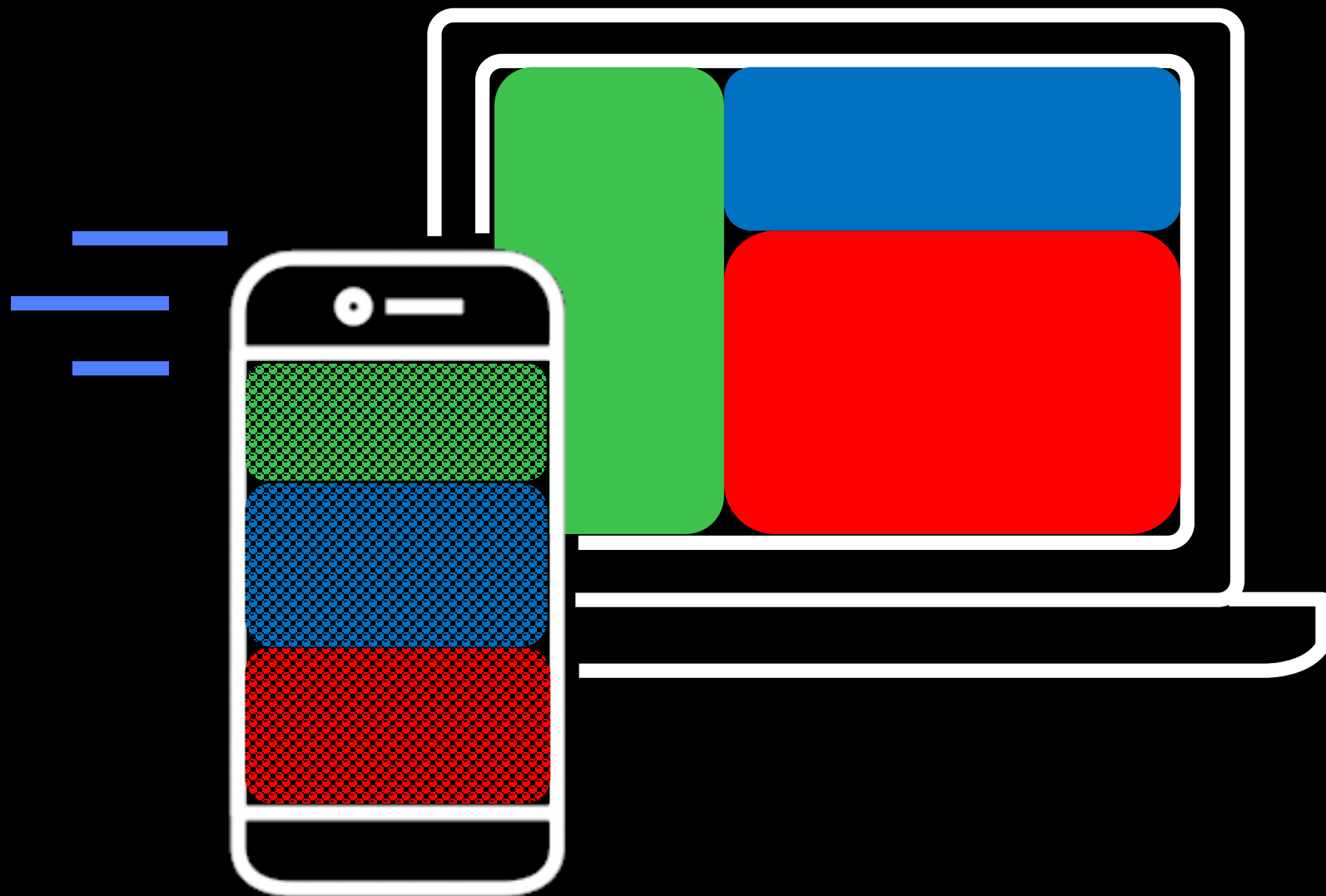
Ask for what you want

```
{  
  project(name: "GraphQL") {  
    tagline  
  }  
}
```

Get predictable results

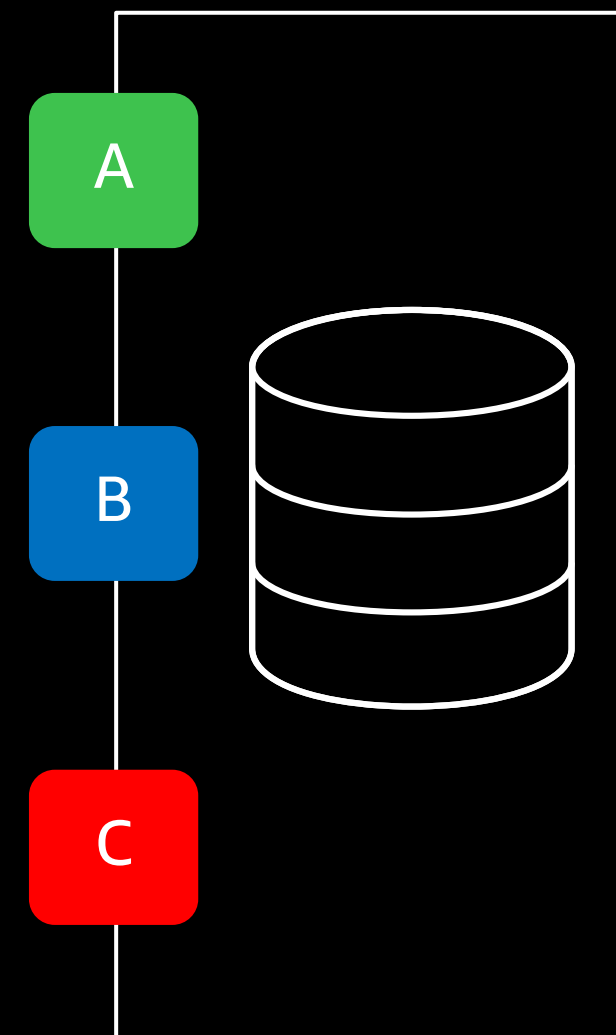
```
{  
  "project": {  
    "tagline": "A query language for APIs"  
  }  
}
```

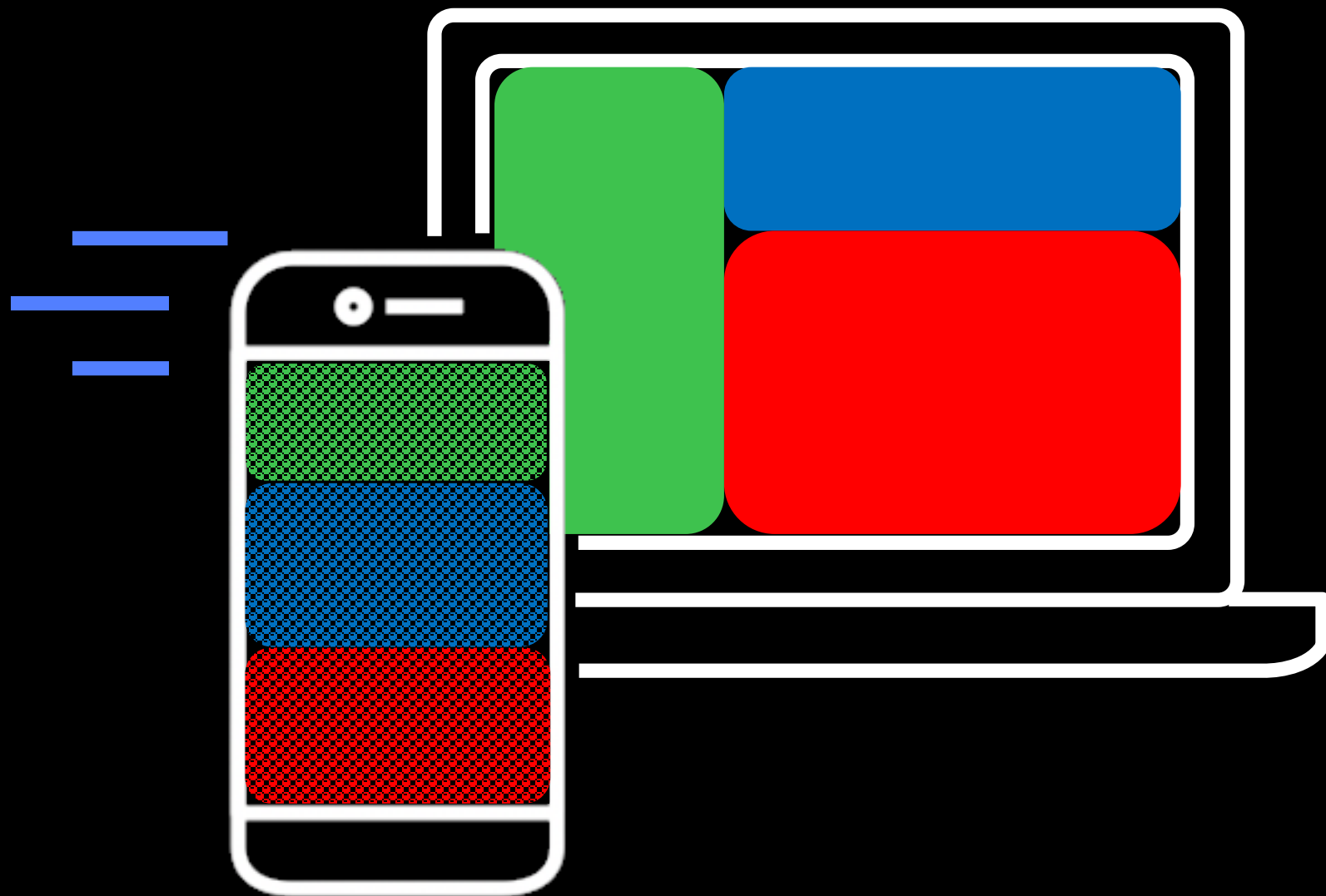




API endpoints

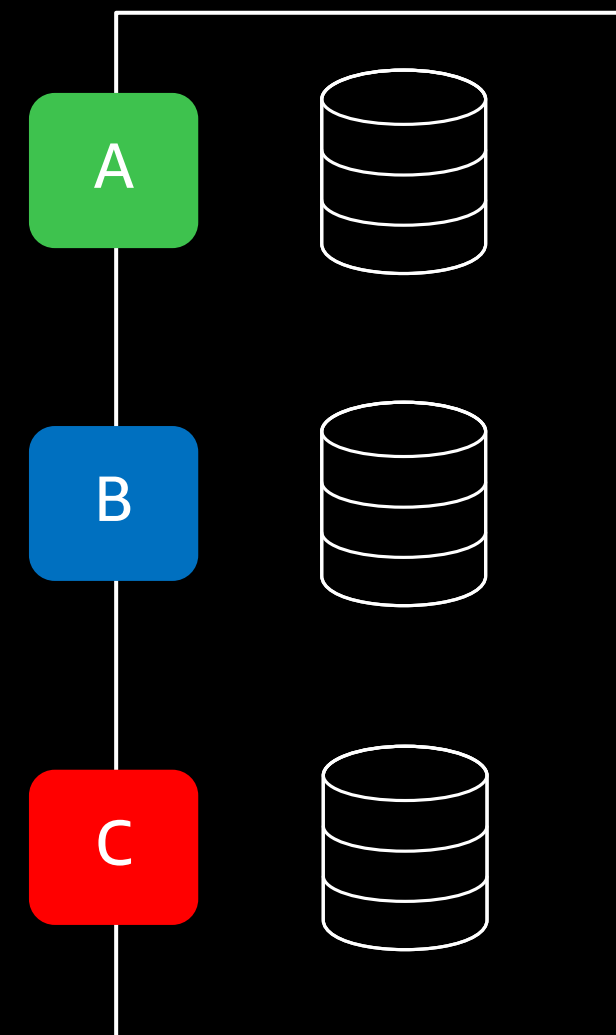
(Data sources)

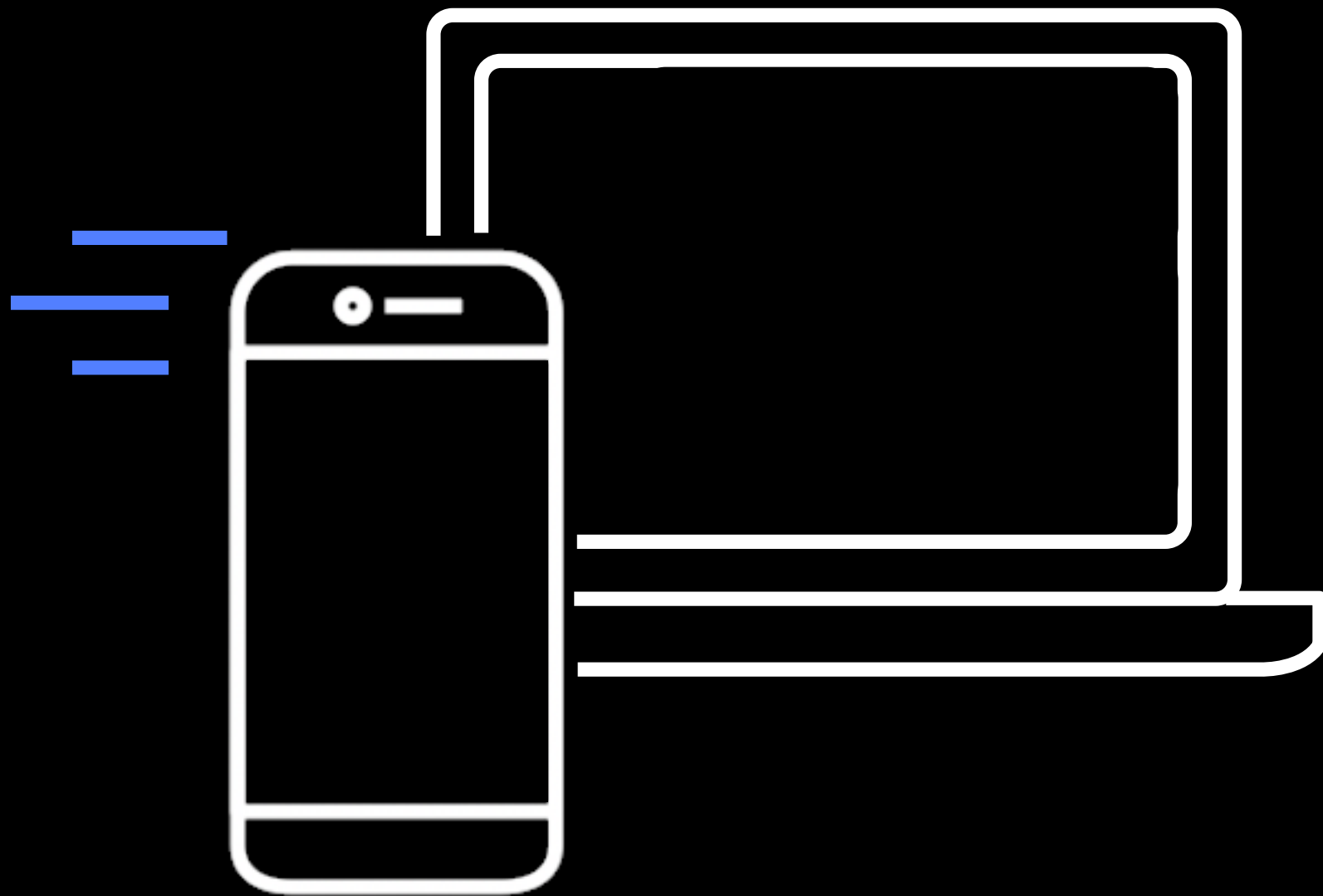




API endpoints

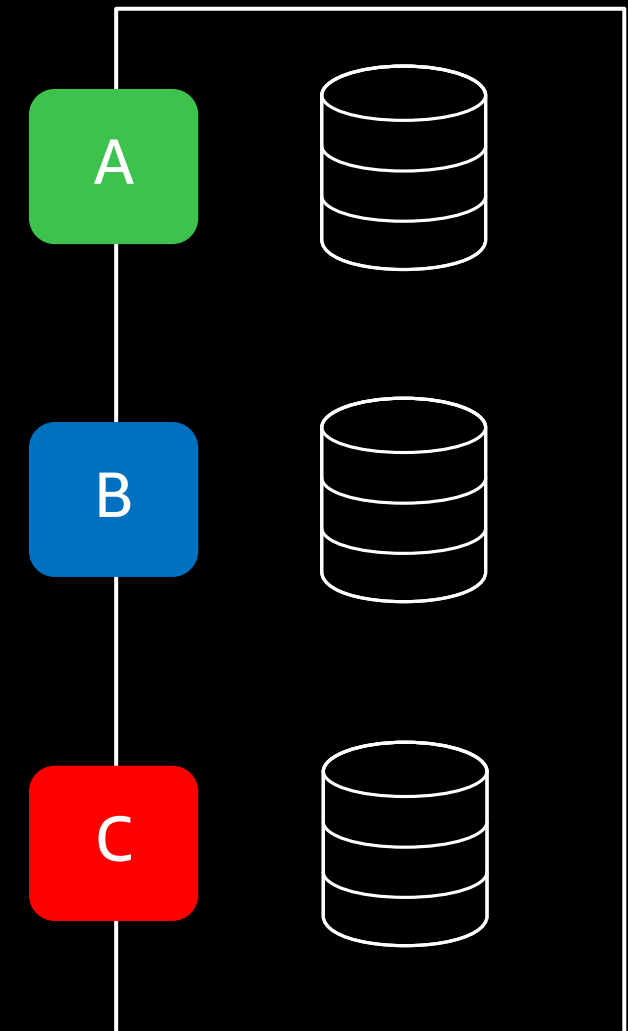
(Data sources)

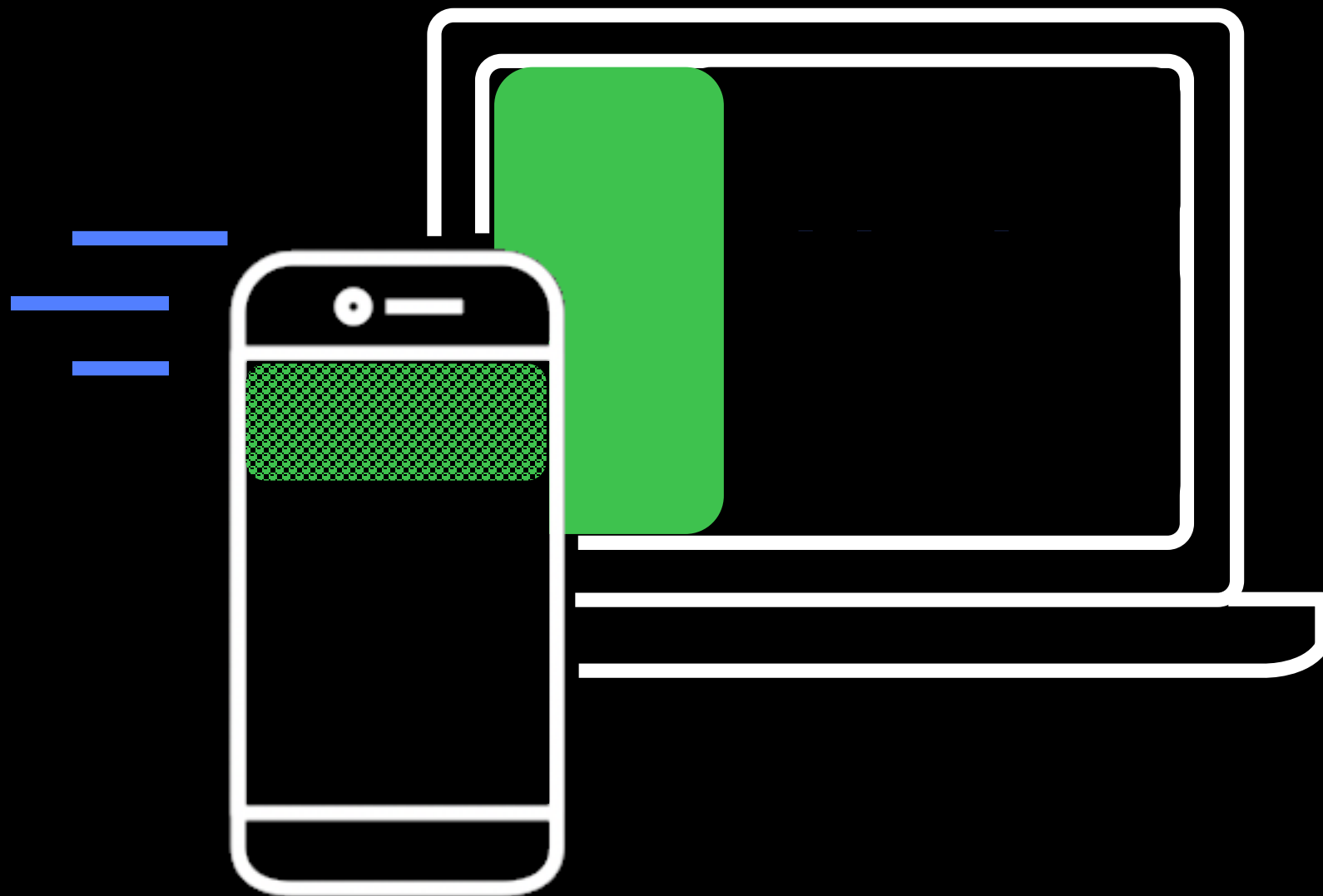




API endpoints

(Data sources)





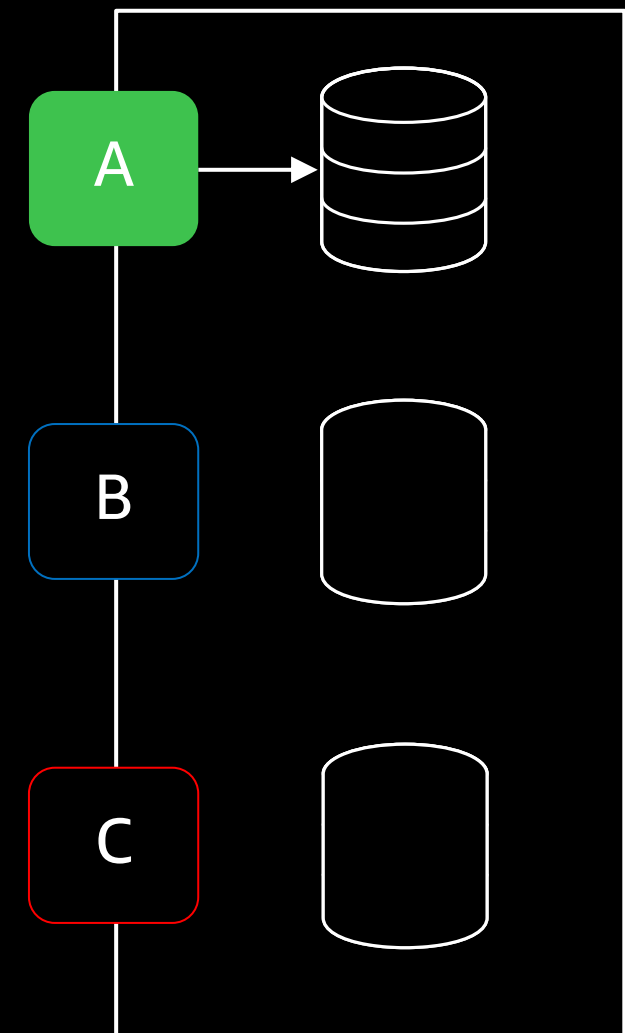
Data payload

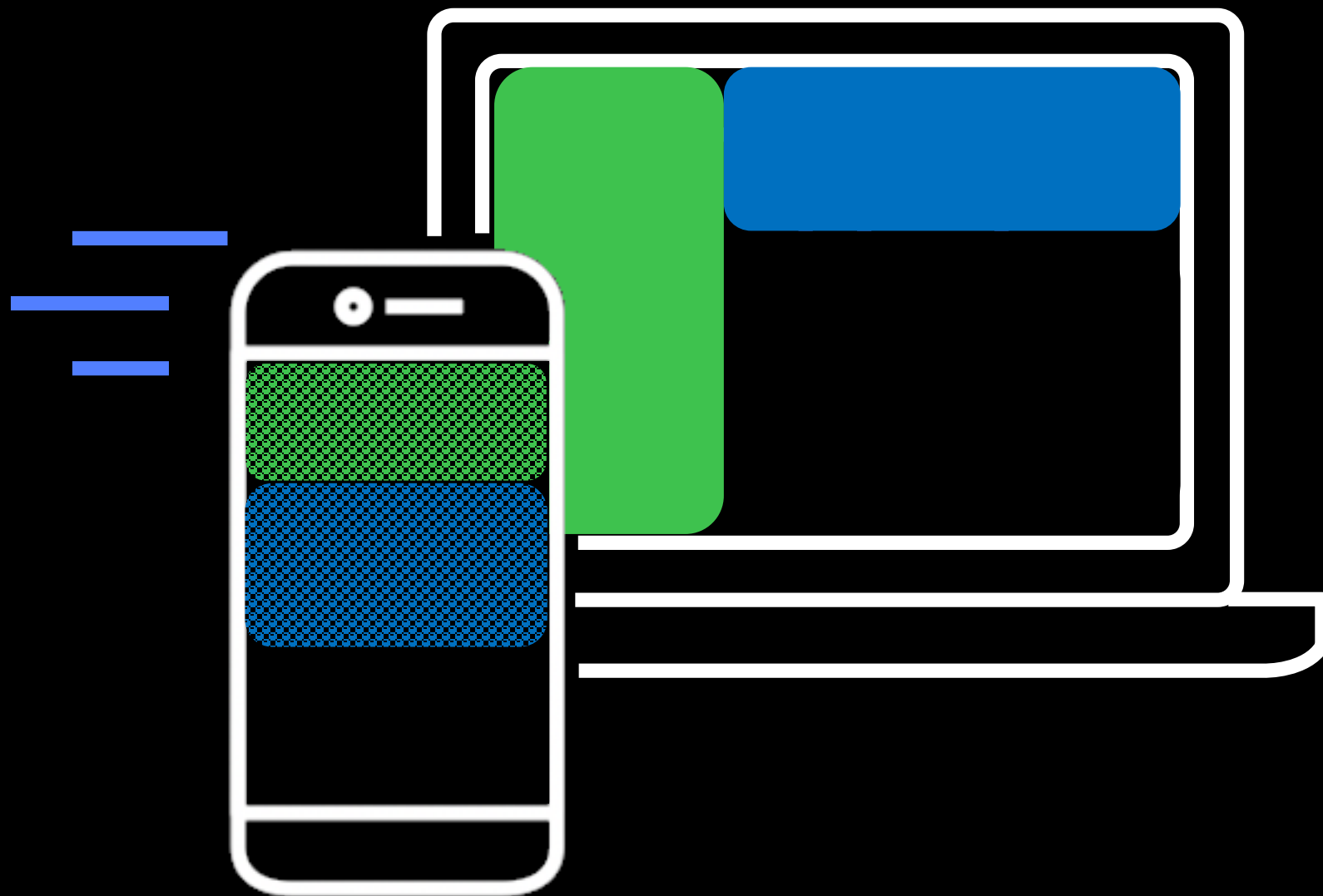
JSON{}



API endpoints

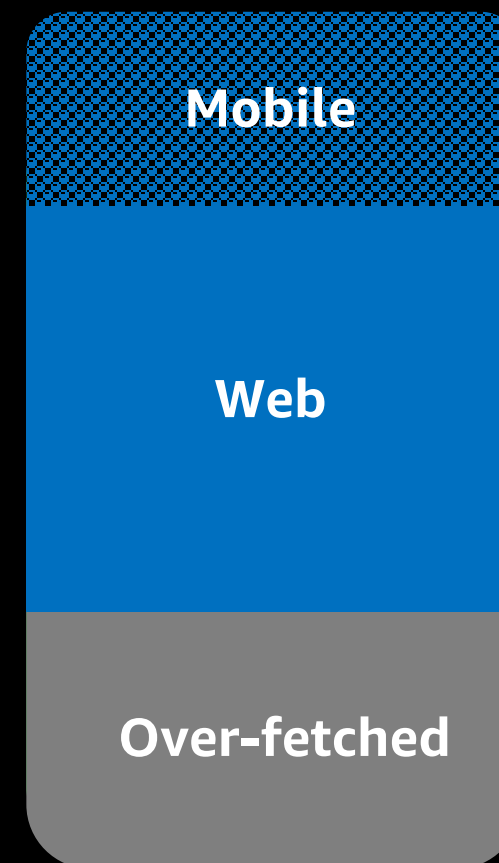
(Data sources)





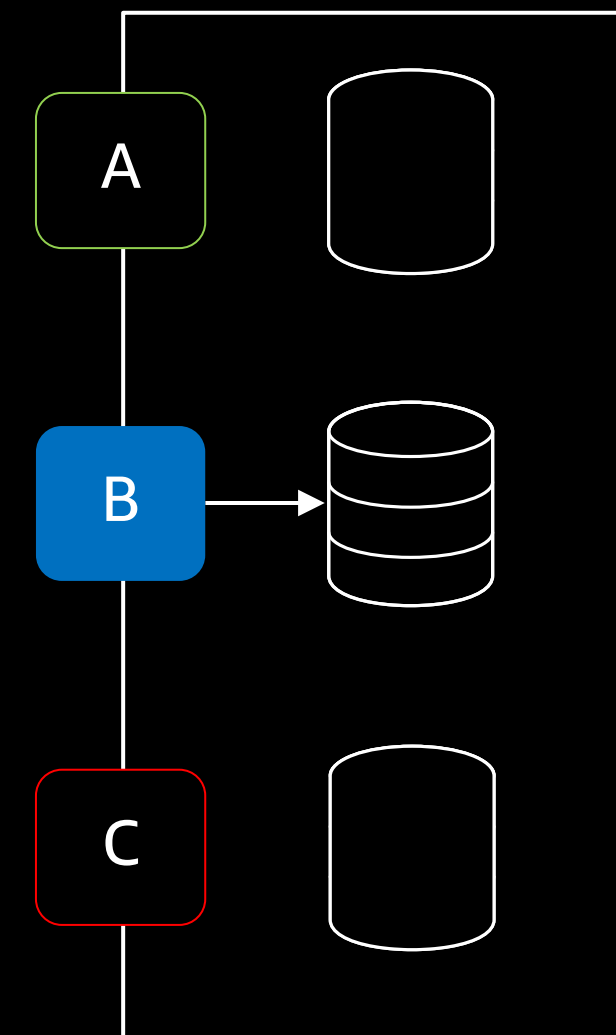
Data payload

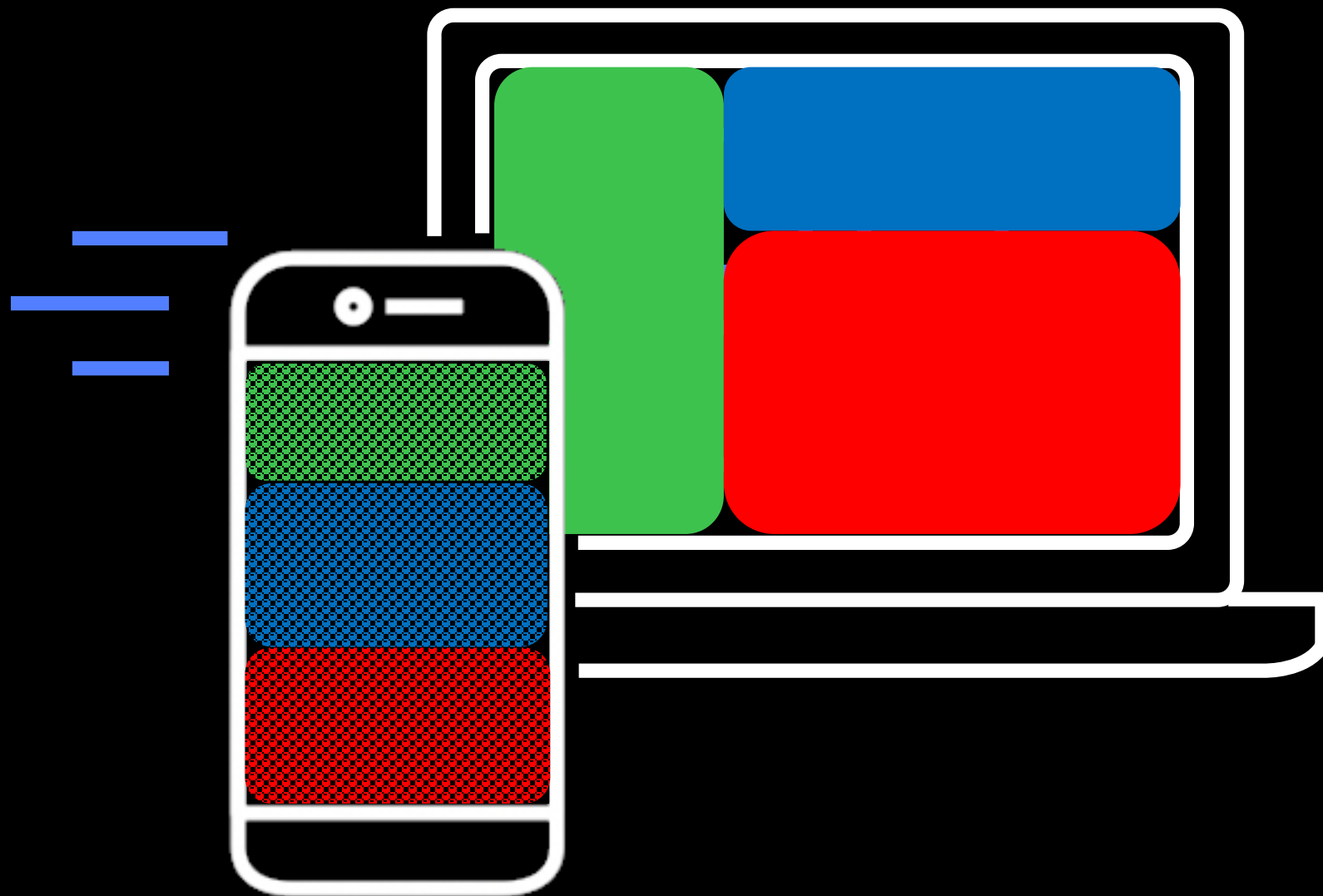
JSON{



API endpoints

(Data sources)





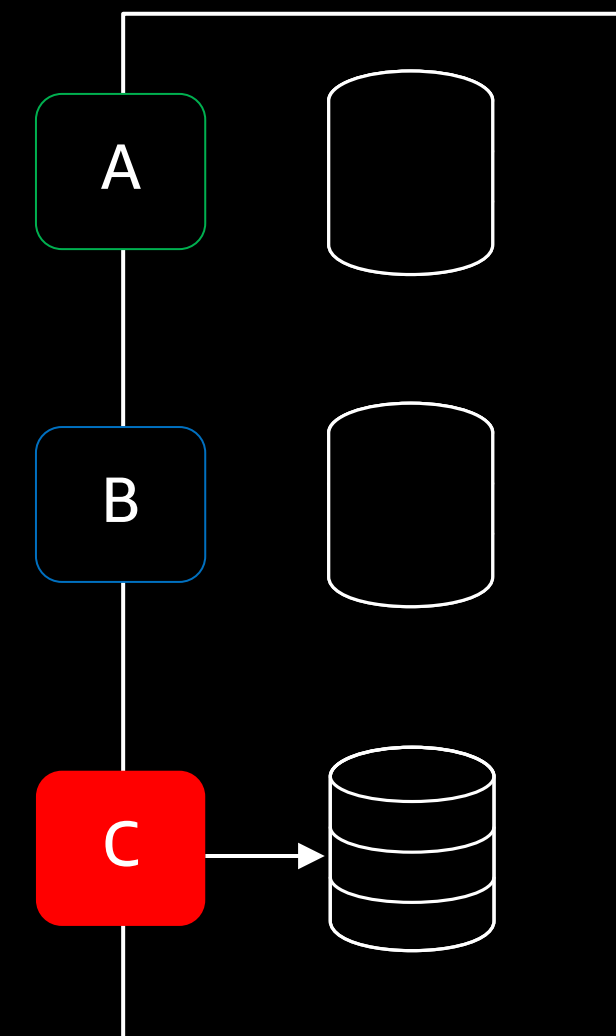
Data payload

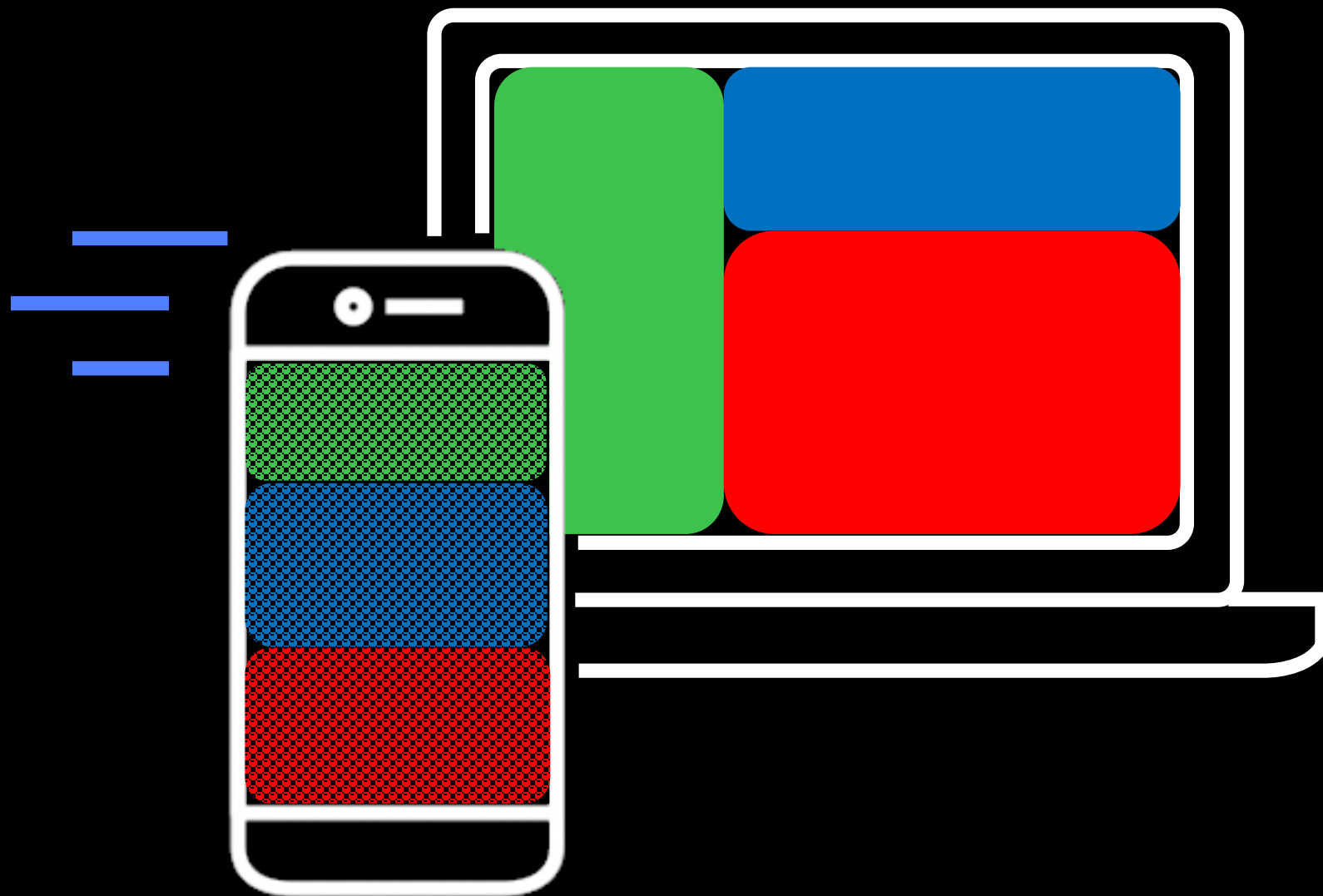
JSON{}



API endpoints

(Data sources)



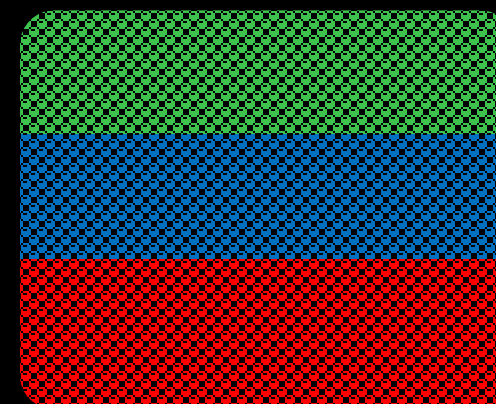


Data payload

JSON{}



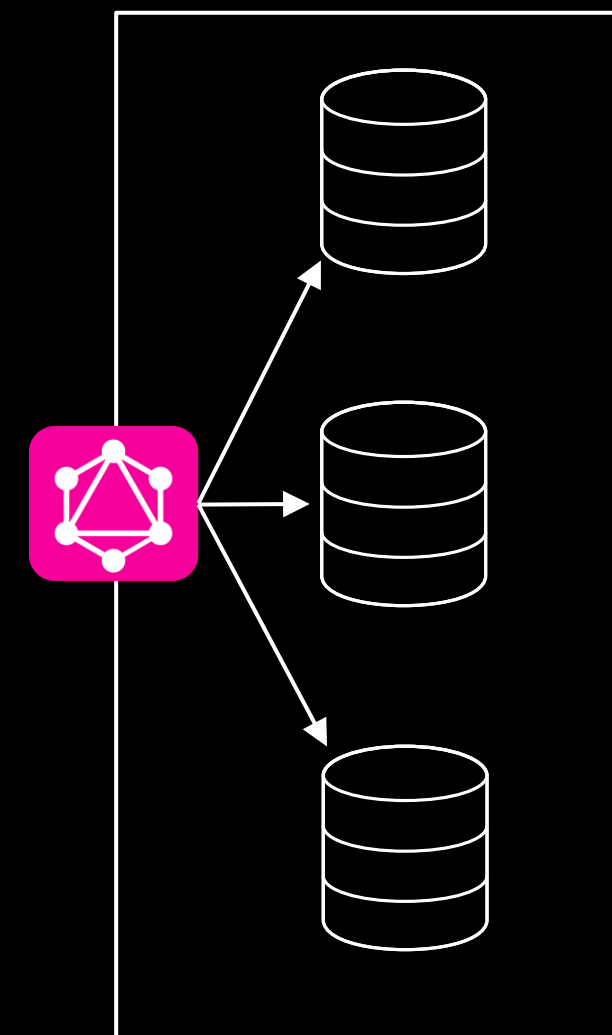
Web

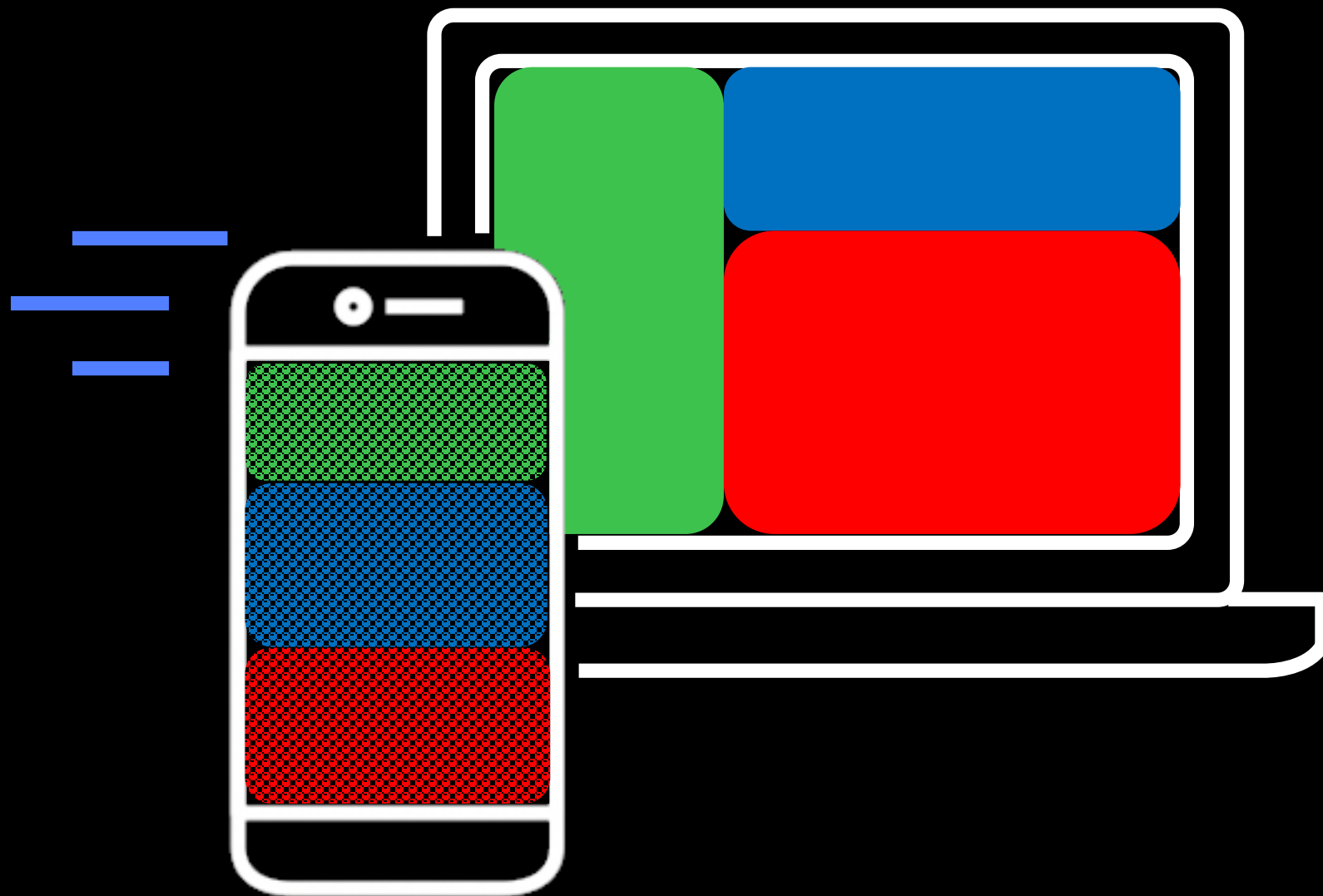


Mobile

API endpoints

(Data sources)



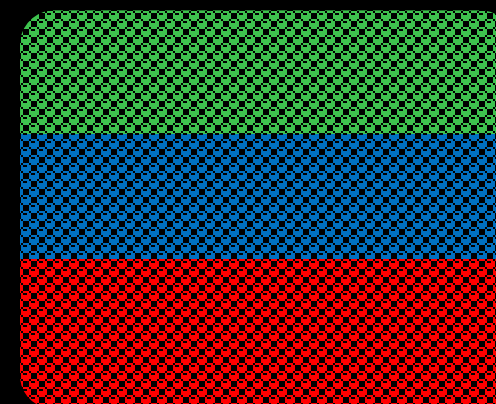


Data payload

JSON{}



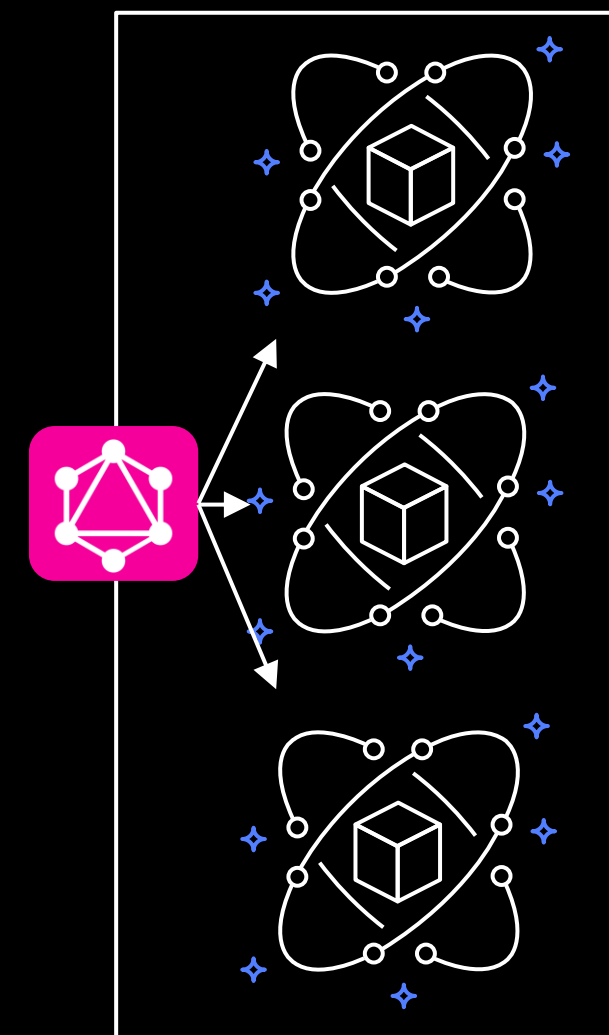
Web

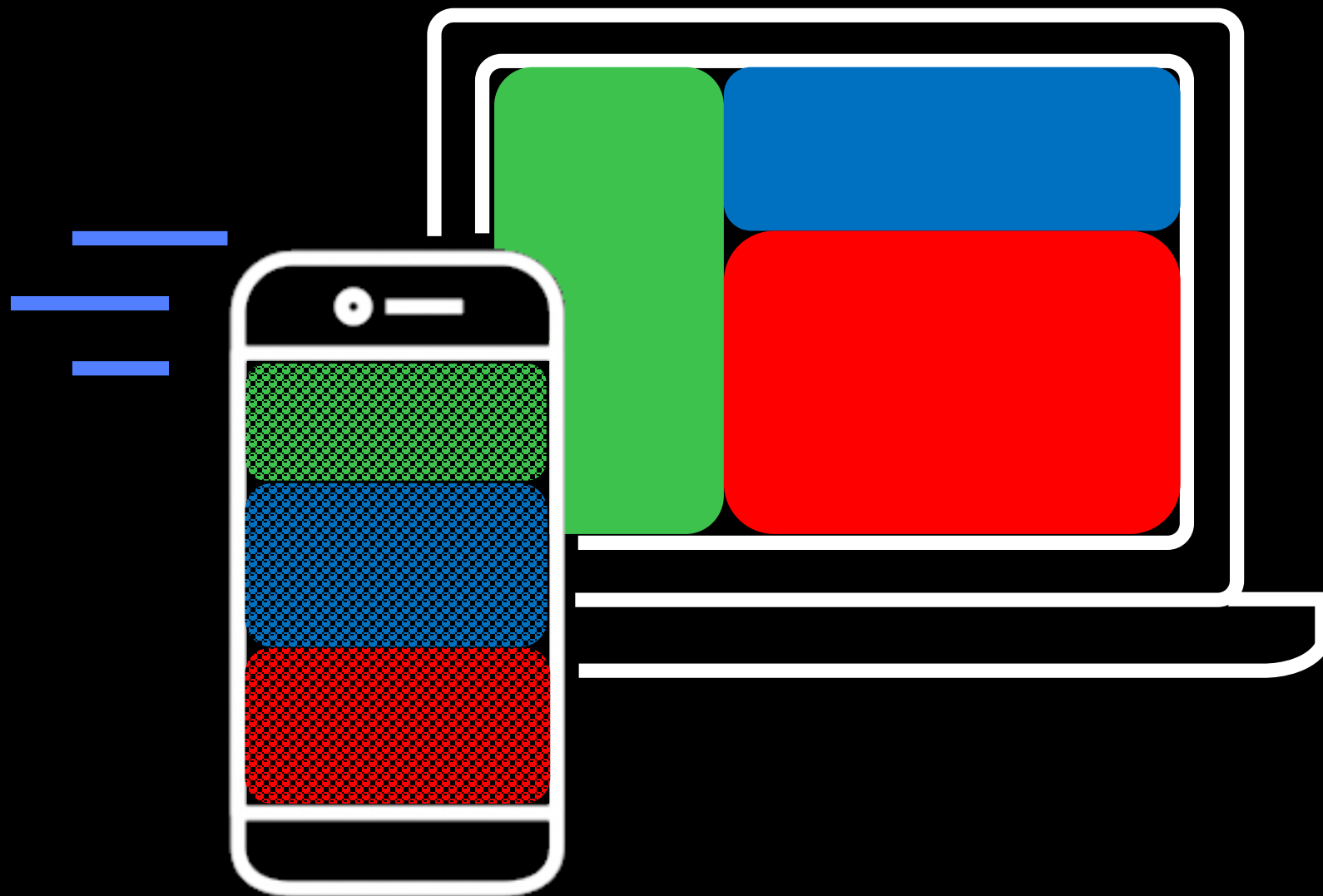


Mobile

API endpoints

(Data sources)



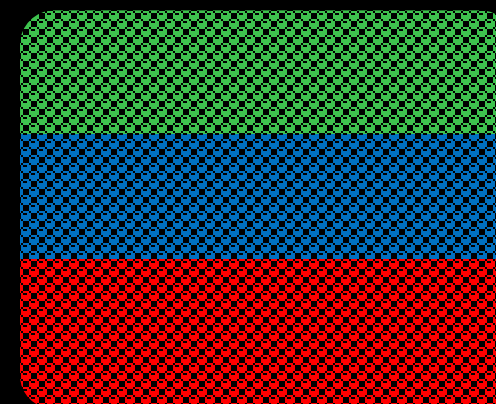


Data payload

JSON{}



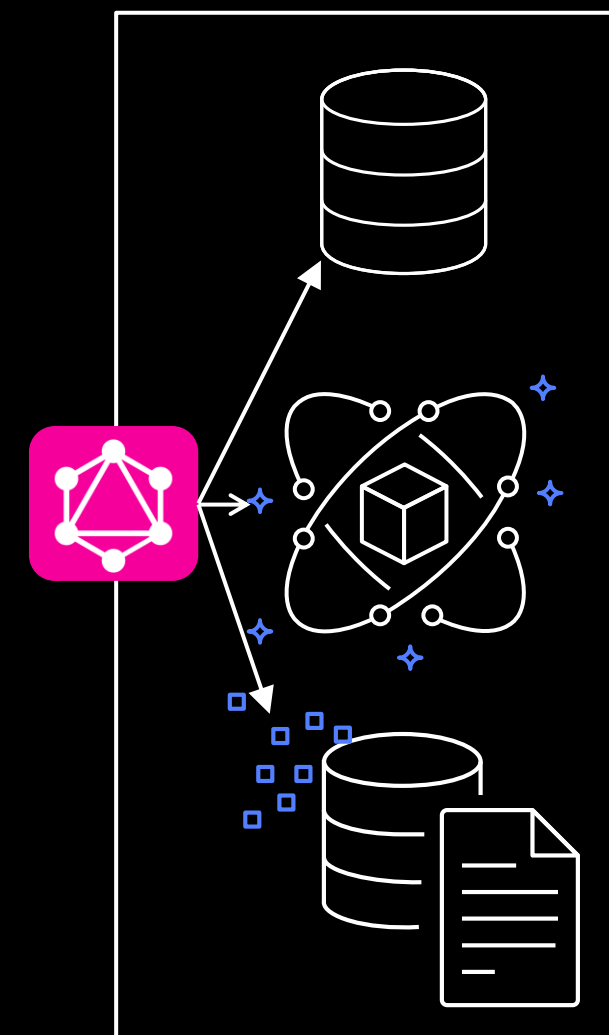
Web



Mobile

API endpoints

(Data sources)

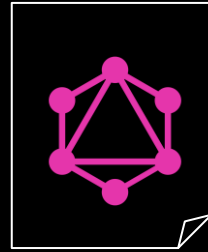


**Get exactly the data you need,
nothing more,
nothing less**

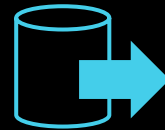


A query language for APIs ... and a runtime

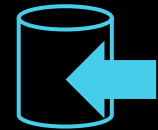
GraphQL schema and operations



Types



Queries



Mutations

Subscriptions

```
type User {  
  id: ID!  
  username: String!  
  firstName: String  
  lastName: String  
  daysActive: Int  
}
```

A query language for APIs ... and a runtime

Queries

```
query GetPost {  
  getPost(id: "1") {  
    id  
    title  
    author  
    date  
  }  
}
```

Mutations

```
mutation CreatePost {  
  createPost(input: {...}) {  
    id  
    title  
    author  
    date  
  }  
}
```

Subscriptions

```
subscription OnCreatePost {  
  onCreatePost {  
    id  
    title  
    author  
    date  
  }  
}
```

A query language for APIs ... and a runtime

Queries

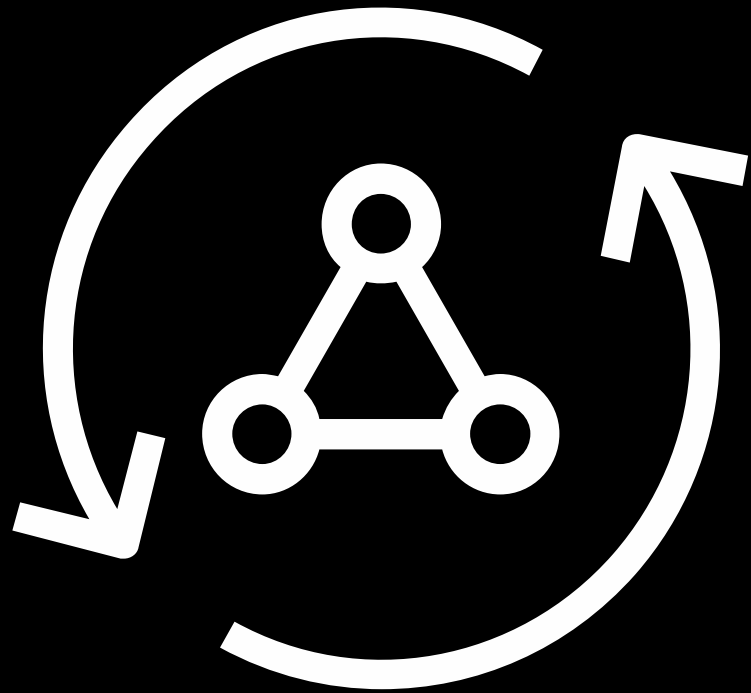
```
query GetPost {  
  getPost(id: "1") {  
    id  
    title  
  }  
}
```

Mutations

```
mutation CreatePost {  
  createPost(input: {...}) {  
    id  
  
    date  
  }  
}
```

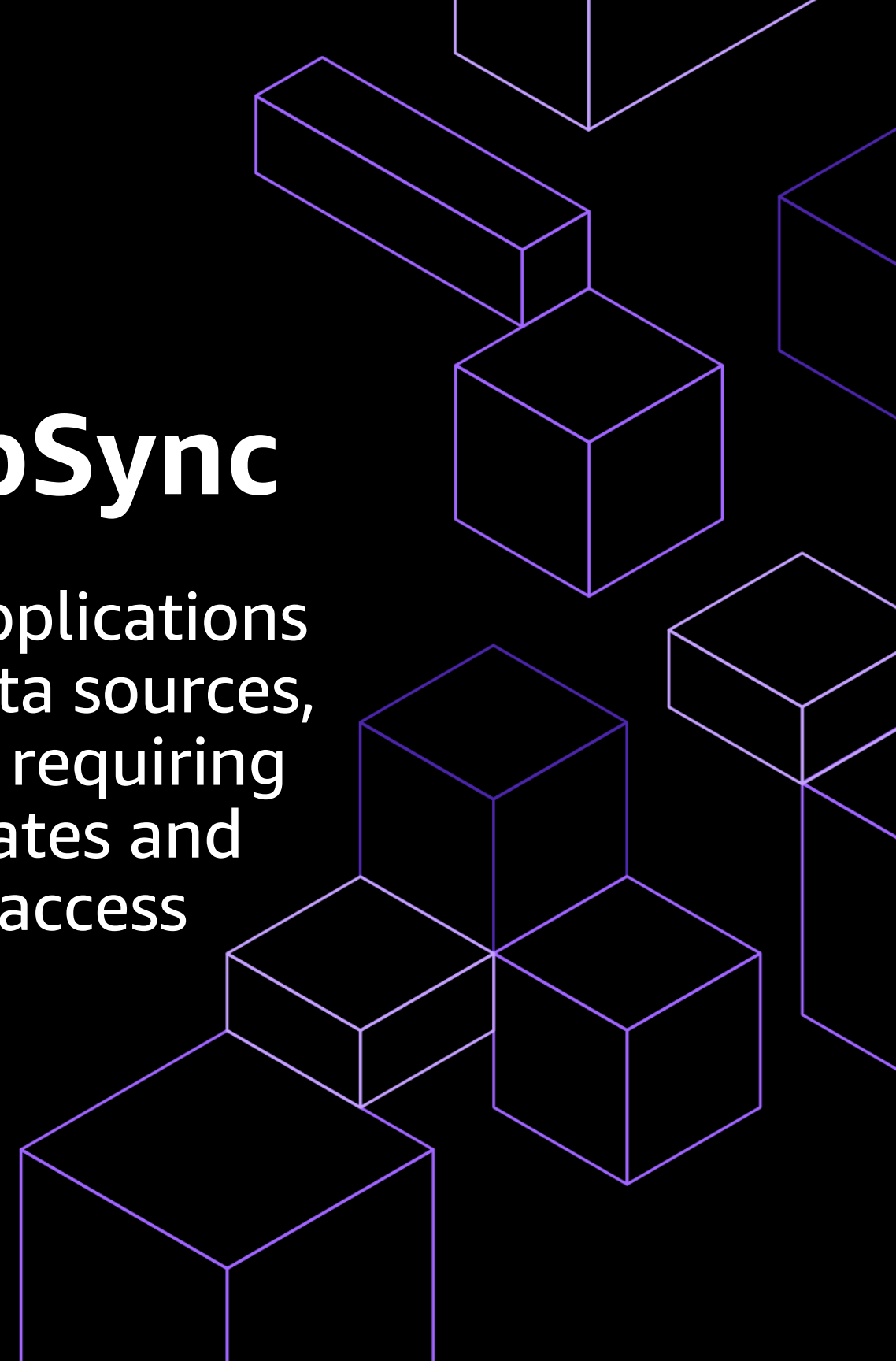
Subscriptions

```
subscription OnCreatePost {  
  onCreatePost {  
    id  
    title  
    author  
  }  
}
```



AWS AppSync

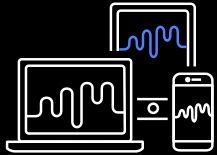
Build scalable applications
on a range of data sources,
including those requiring
real-time updates and
offline data access



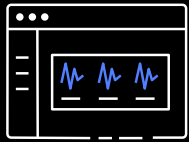
How does AWS AppSync work?



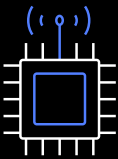
Enterprise apps



Web/mobile apps



Real-time dashboards

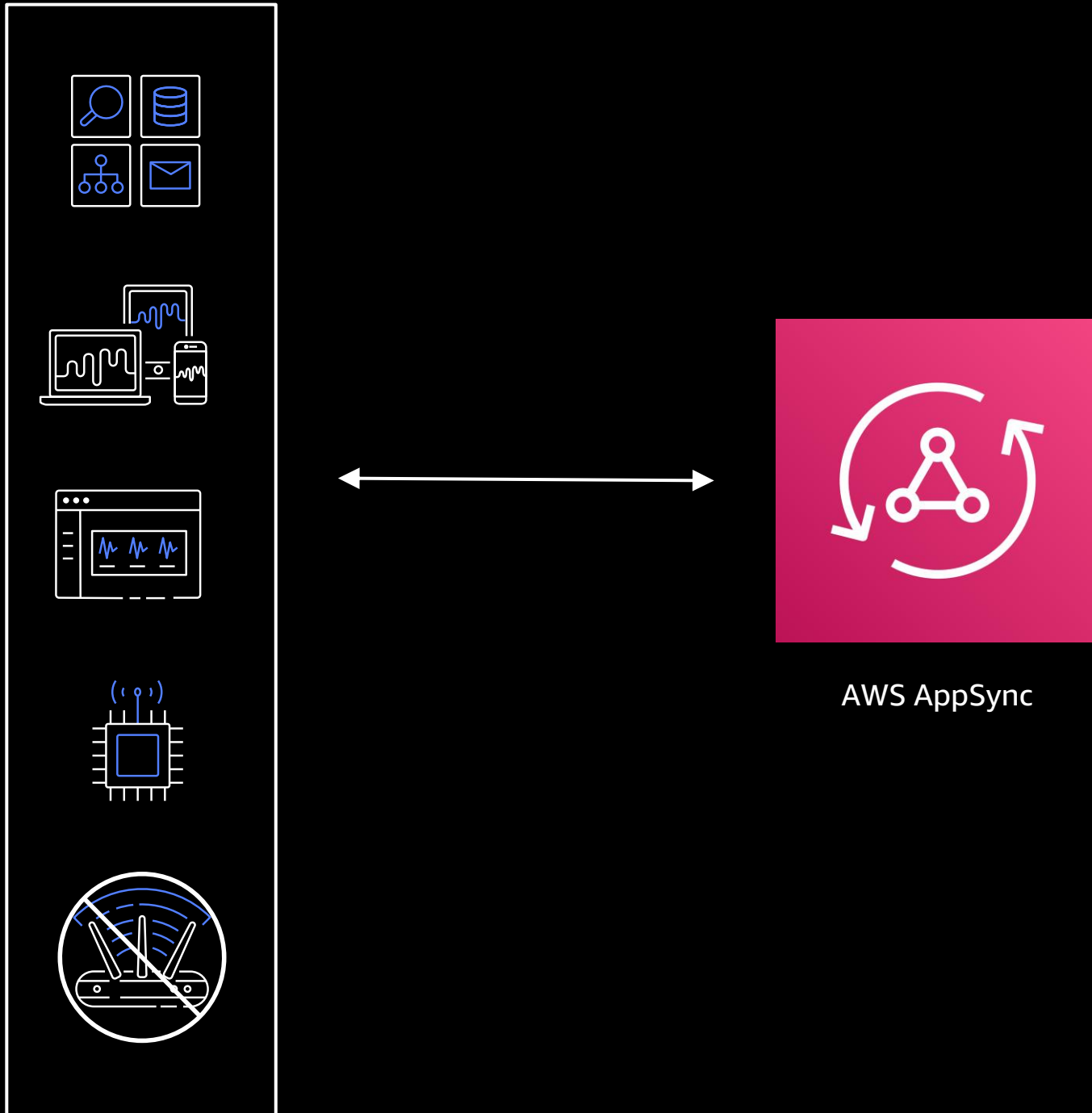


IoT apps

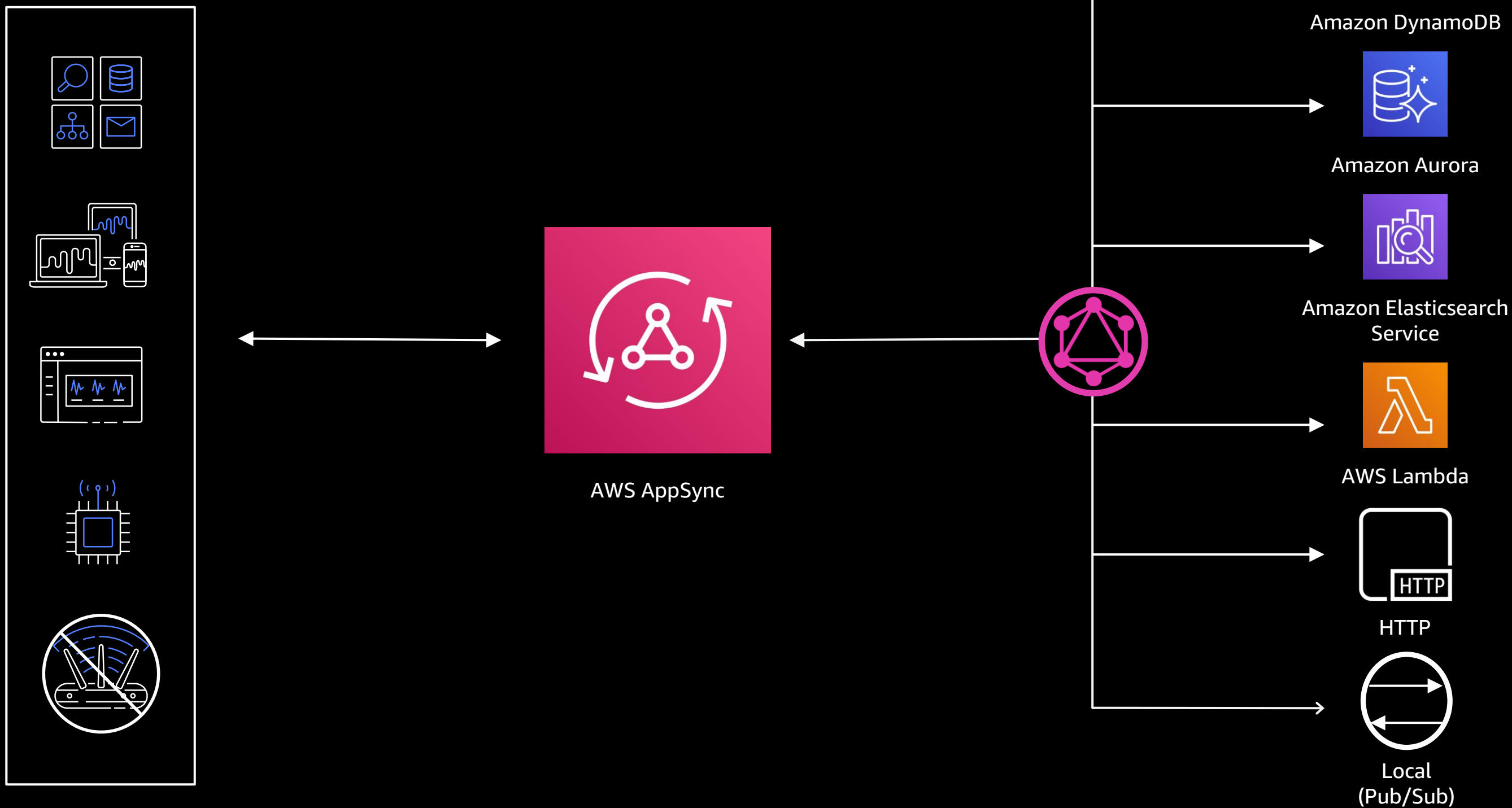


Offline/Delta Sync

How does AWS AppSync work?



How does AWS AppSync work?

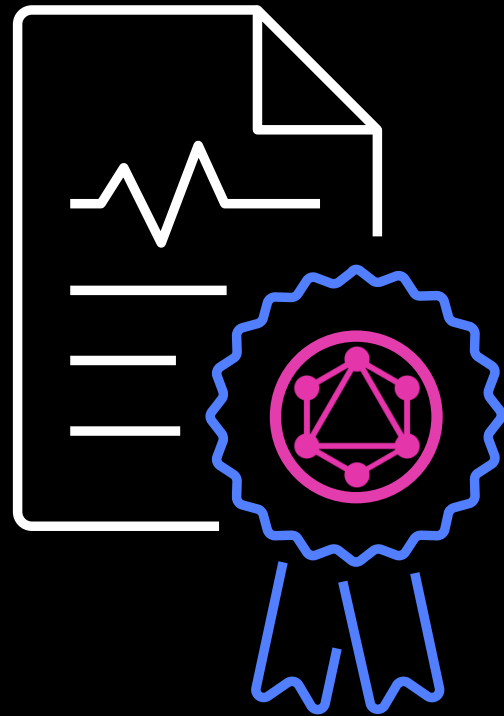


AWS AppSync

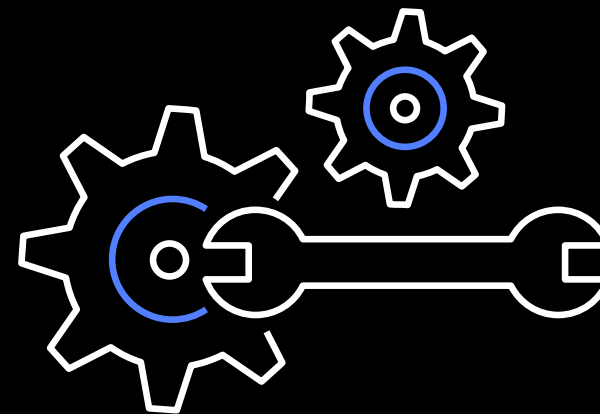
GraphQL API



AWS AppSync



GraphQL Schema
(Data model)



Resolvers
(Business logic)

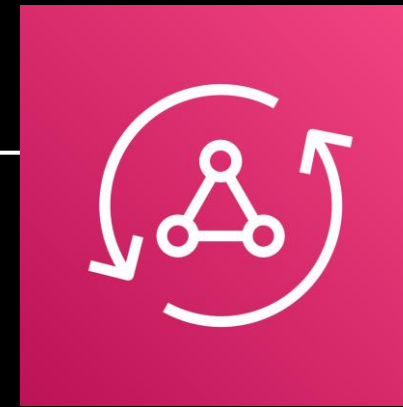


Data sources
(Data)

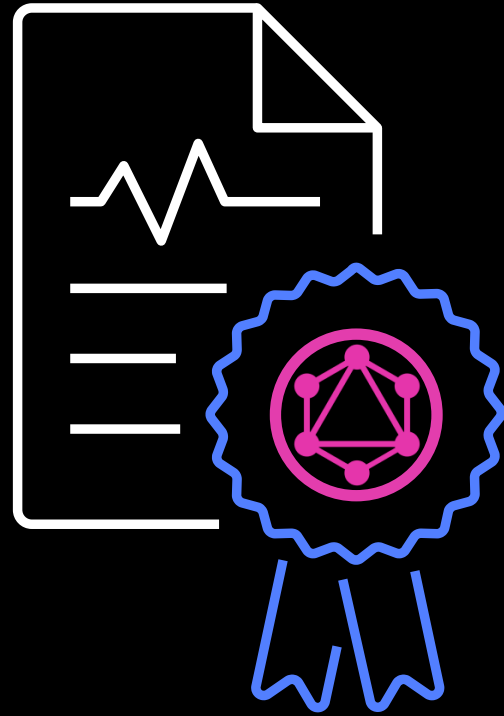


AWS AppSync

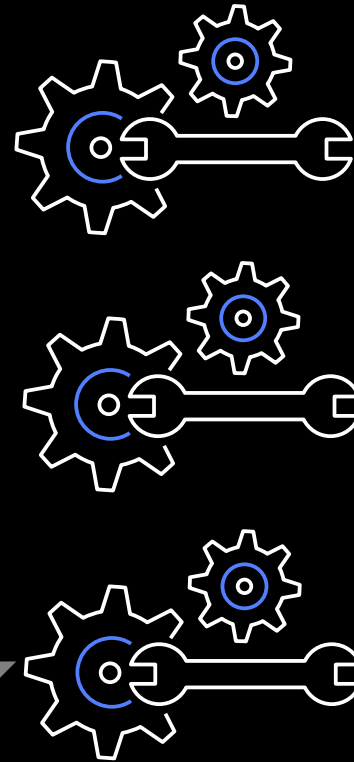
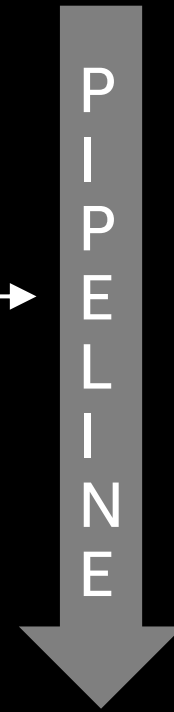
GraphQL API



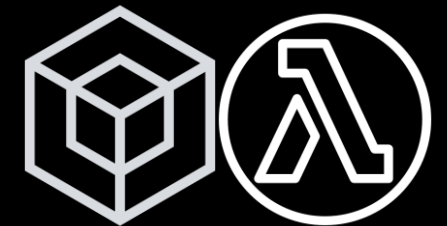
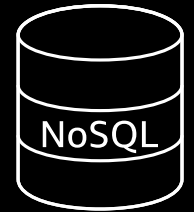
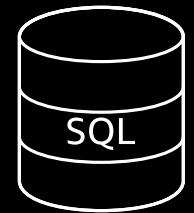
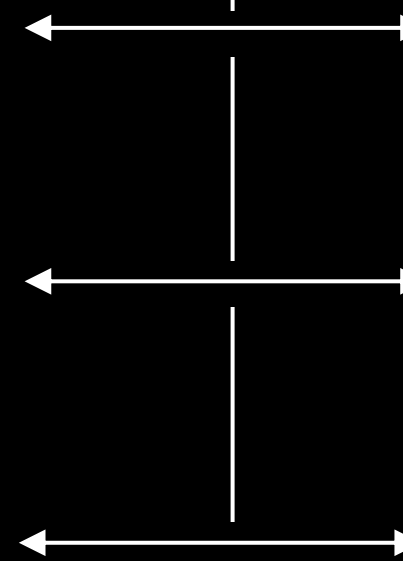
AWS AppSync



GraphQL Schema
(Data model)



Resolvers
(Business logic)

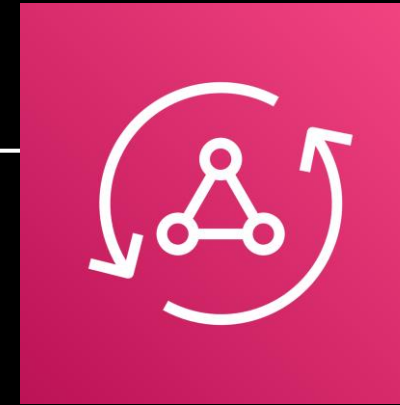


Data sources
(Data)

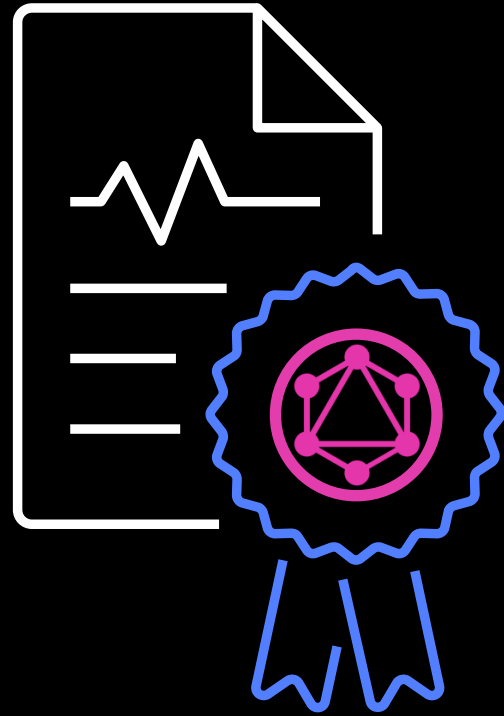


AWS AppSync – Direct Lambda

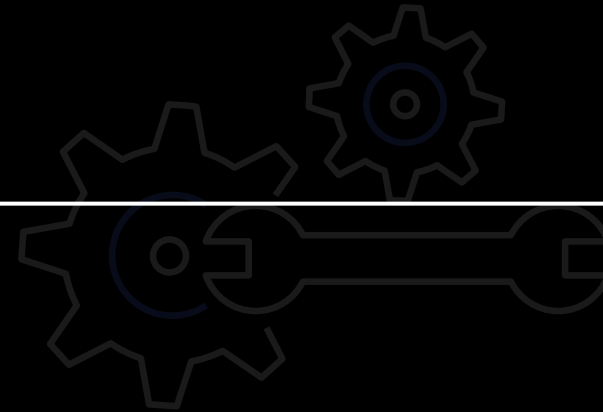
GraphQL API



AWS AppSync



GraphQL Schema
(Data model)

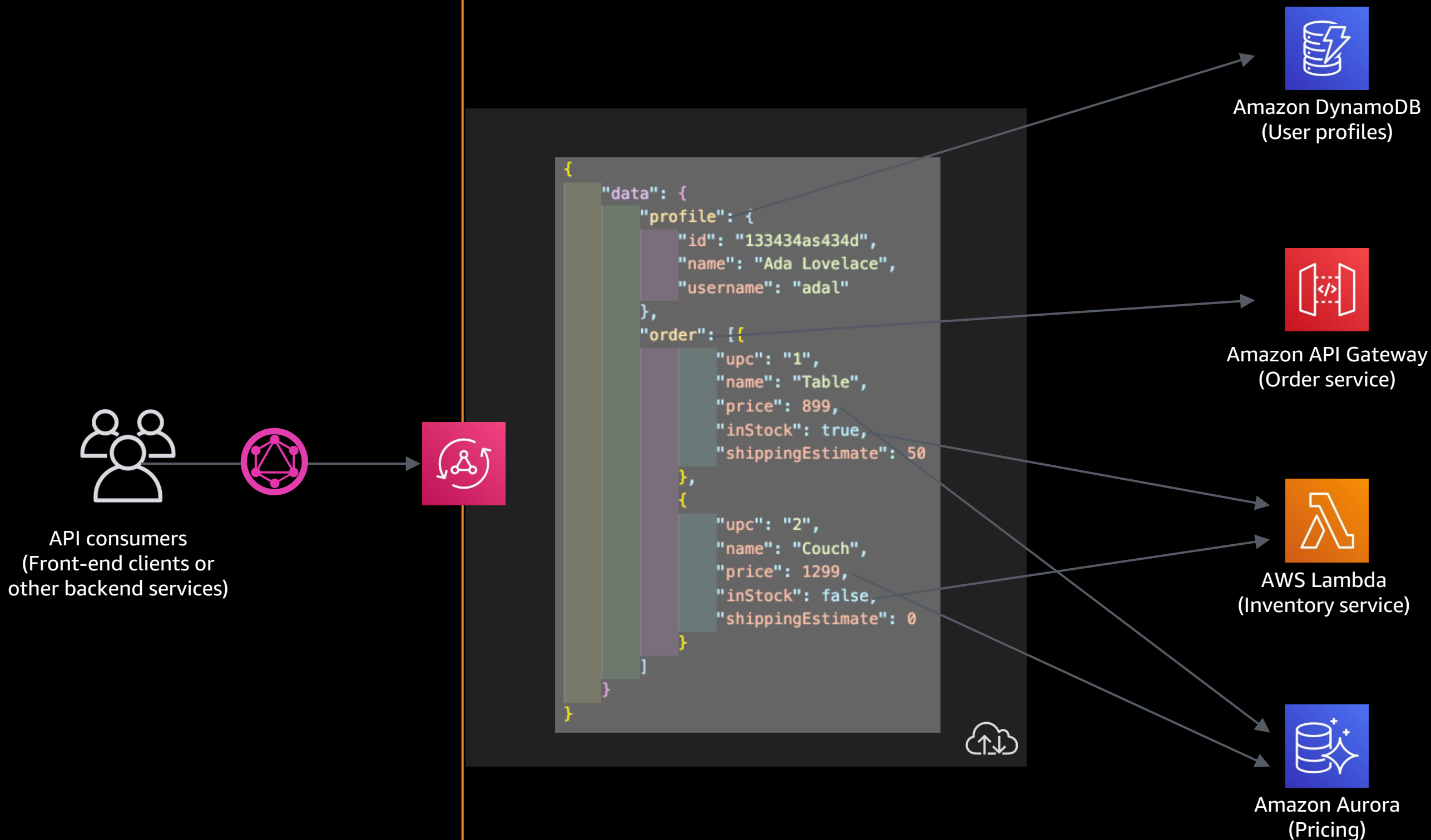


Resolvers
(Business logic)



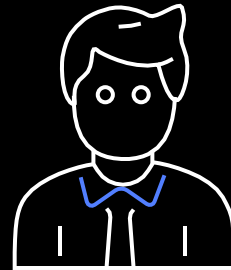
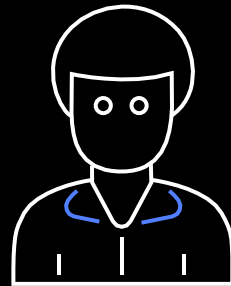
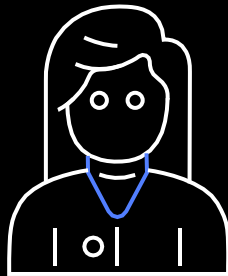
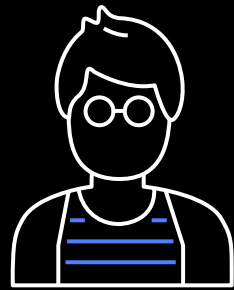
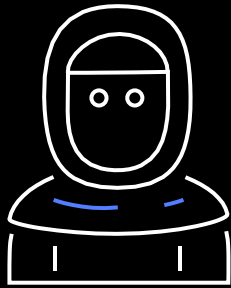
AWS Lambda
(Business logic + Data)



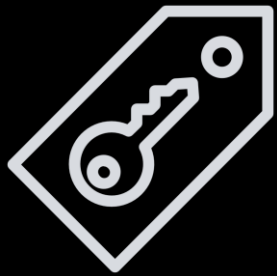


Single request
Single endpoint

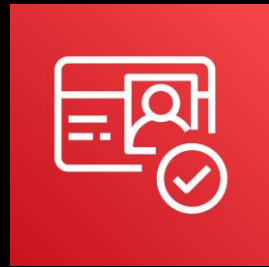
Backend complexity



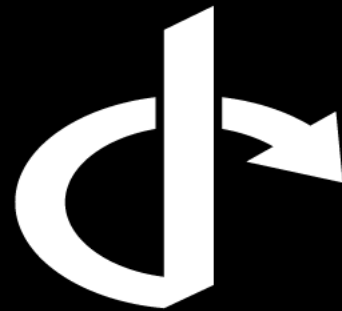
Security



API keys



Amazon Cognito
User Pools



OpenID Connect



AWS Identity and
Access Management

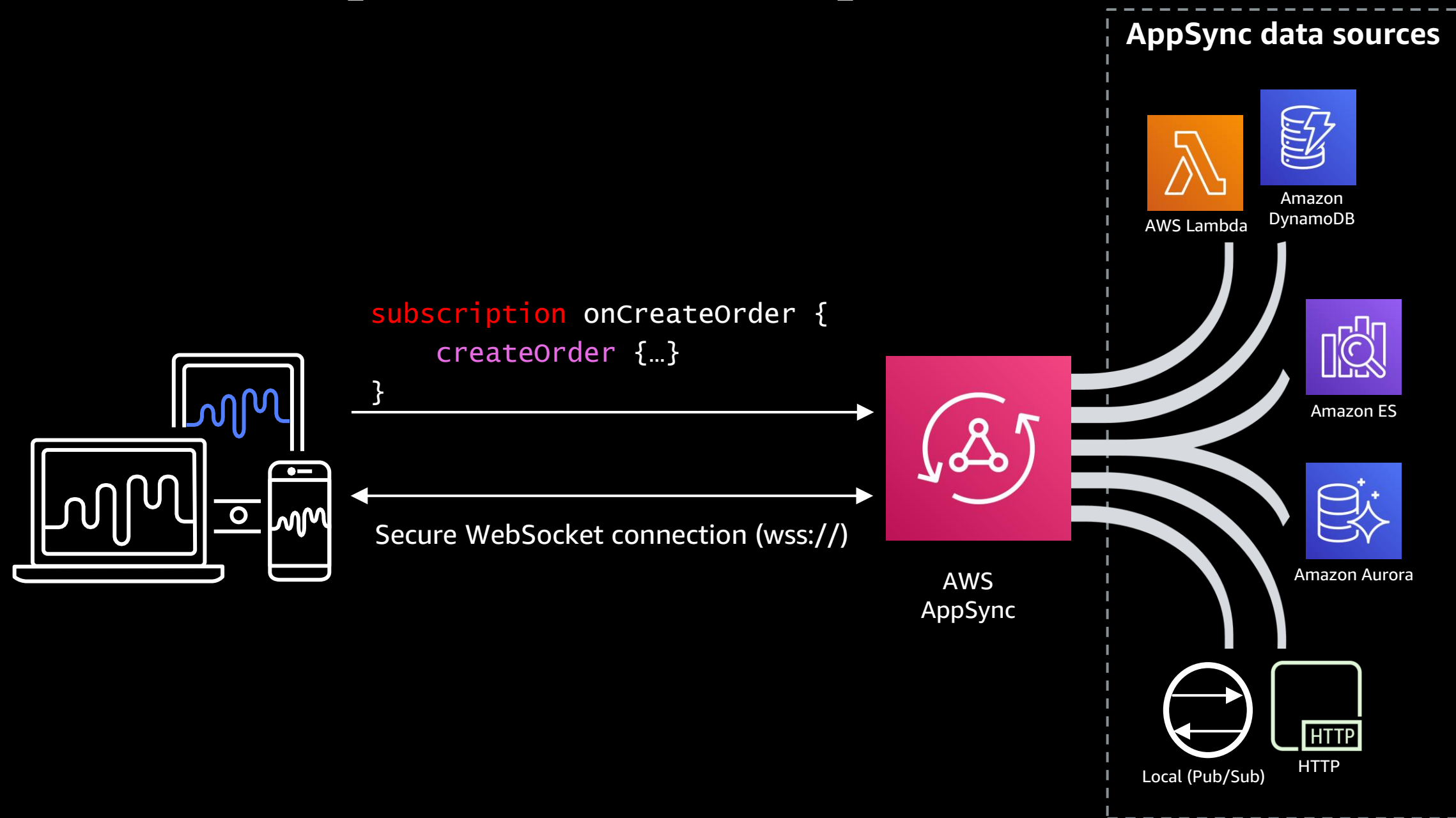


AWS Lambda*

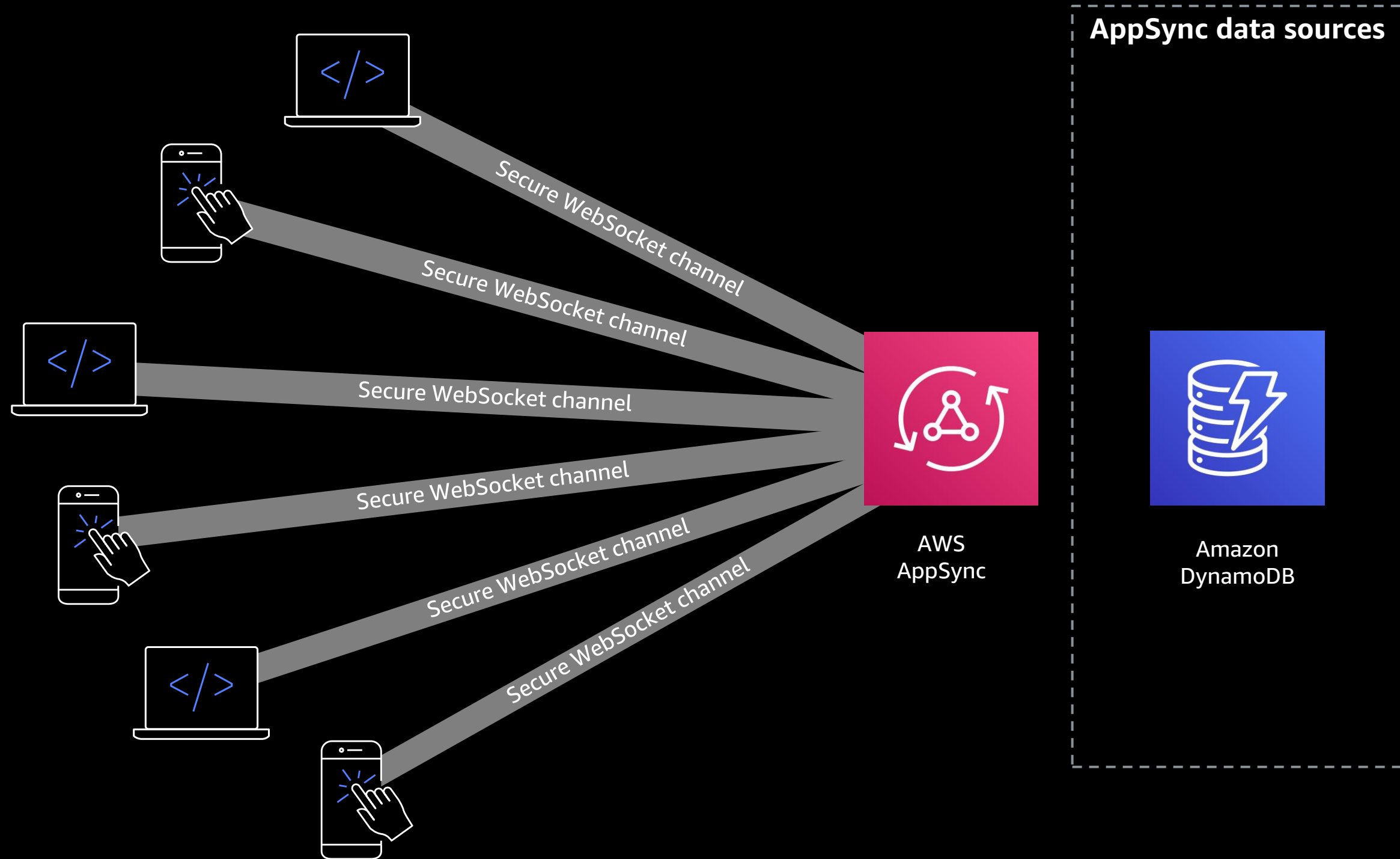
Multi-Auth

```
type Flight
  @aws_api_key
  @aws_cognito_user_pools (cognito_groups: ["FrequentFlyers"])
  @aws_iam
  {
    id: ID! @aws_api_key @aws_iam
    departureDate: String!
    departureAirportCode: String! @aws_api_key
    departureAirportName: String! @aws_api_key
    departureCity: String!
    departureLocale: String!
    arrivalDate: String!
    arrivalAirportCode: String! @aws_iam
    arrivalAirportName: String! @aws_iam
    arrivalCity: String! @aws_iam
    arrivalLocale: String! @aws_iam
    ticketPrice: Int!
    ticketCurrency: String!
    flightNumber: Int!
    seatAllocation: Int
    seatCapacity: Int!
  }
```

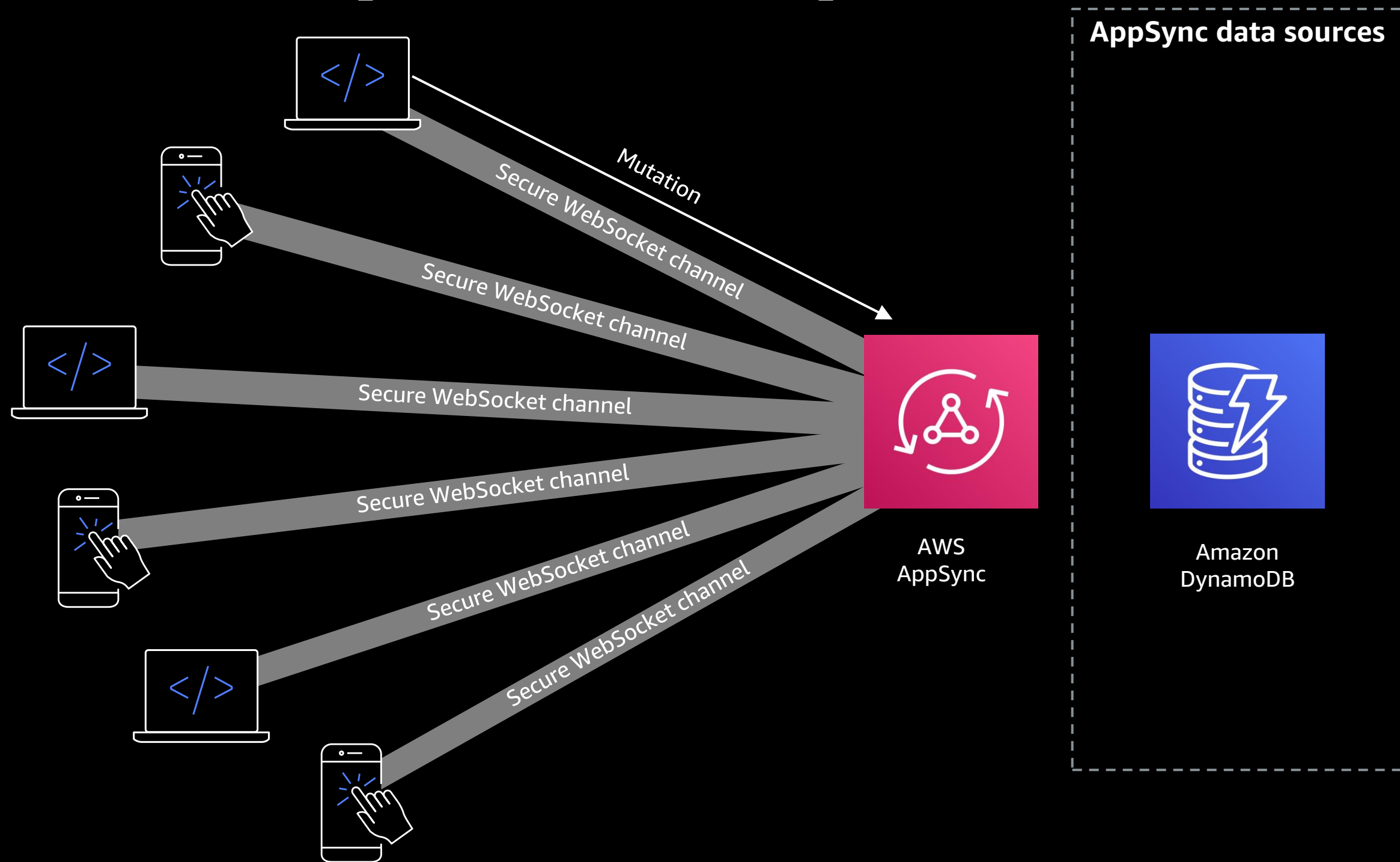
Real-time GraphQL subscriptions



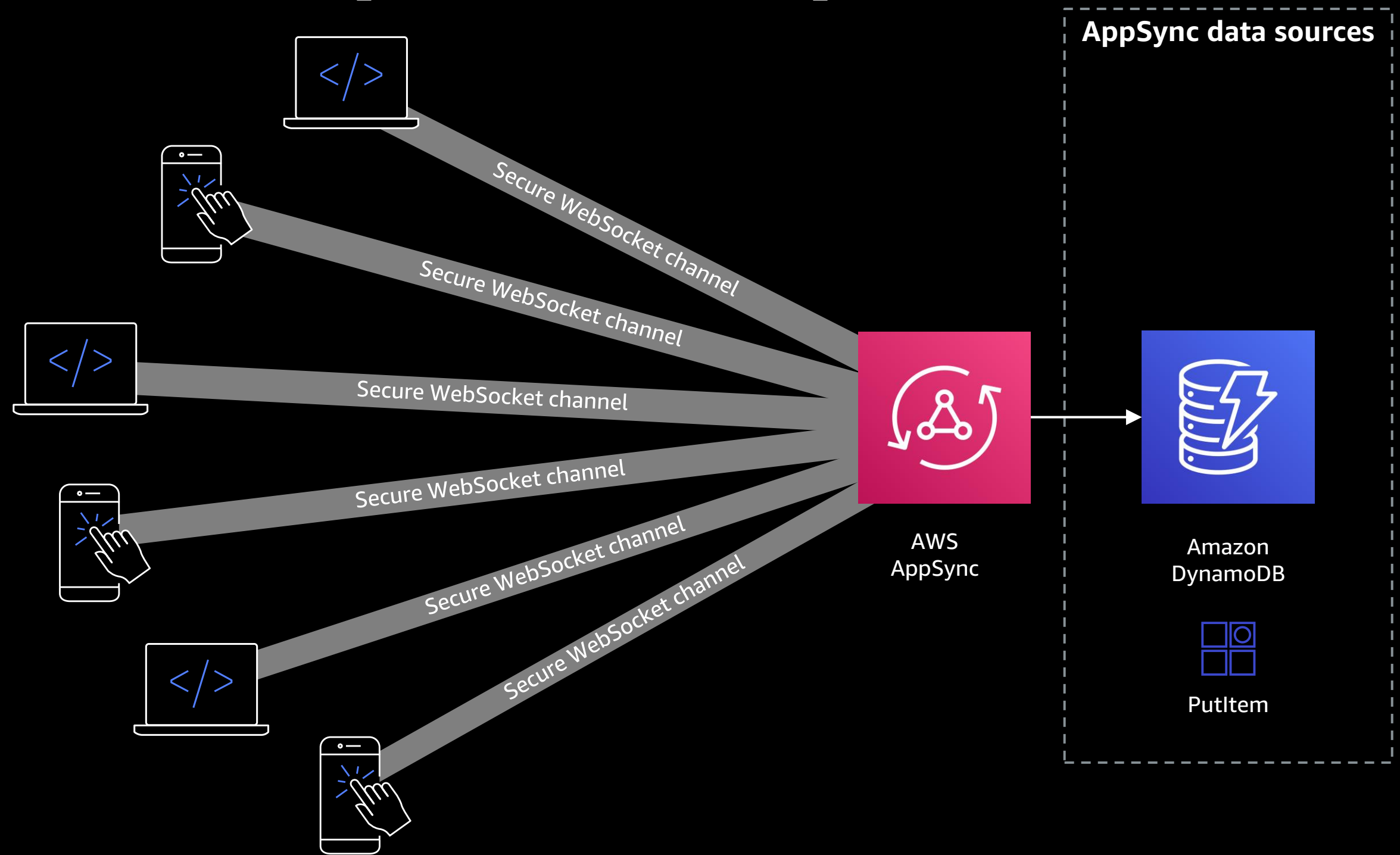
Real-time GraphQL subscriptions



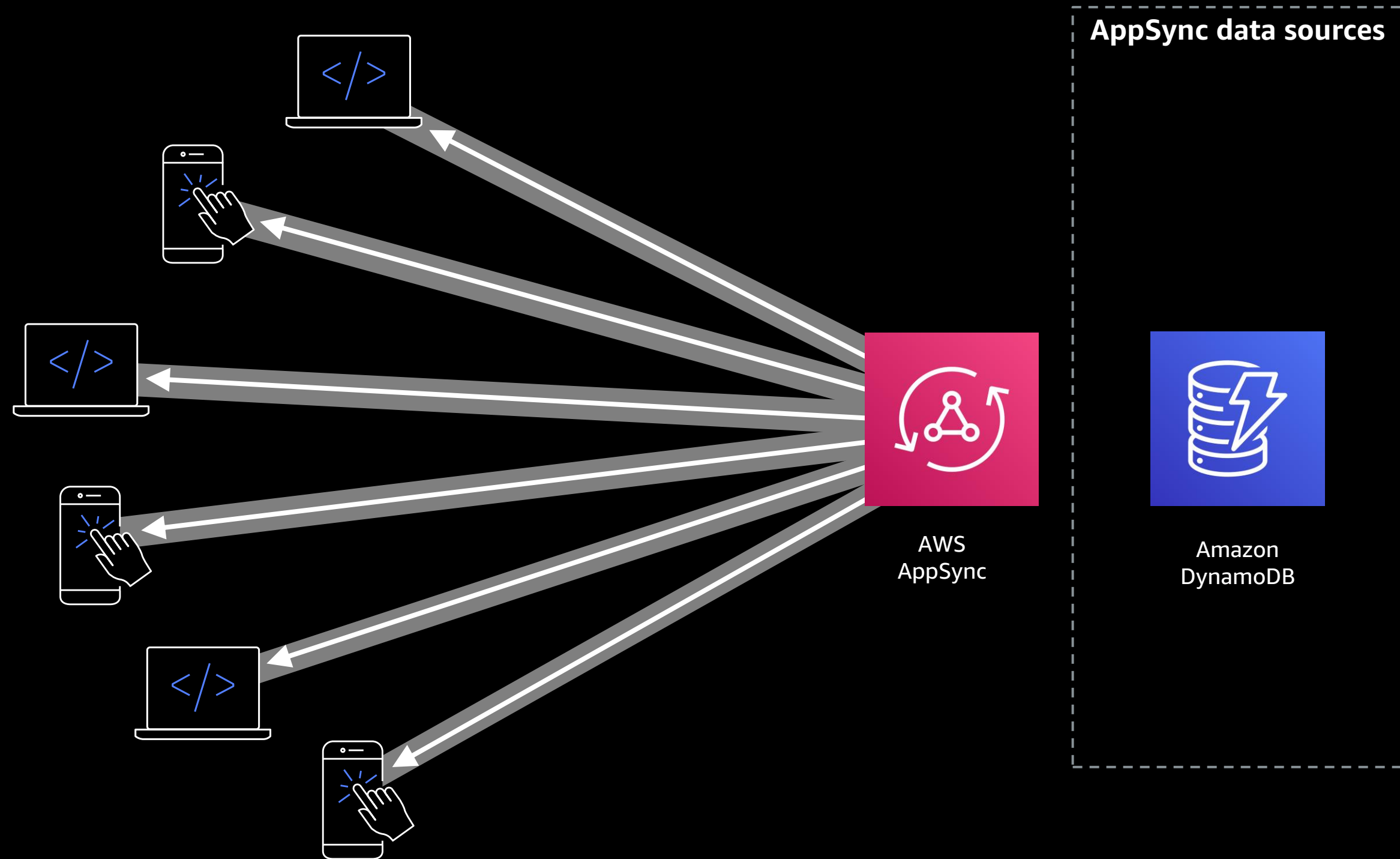
Real-time GraphQL subscriptions



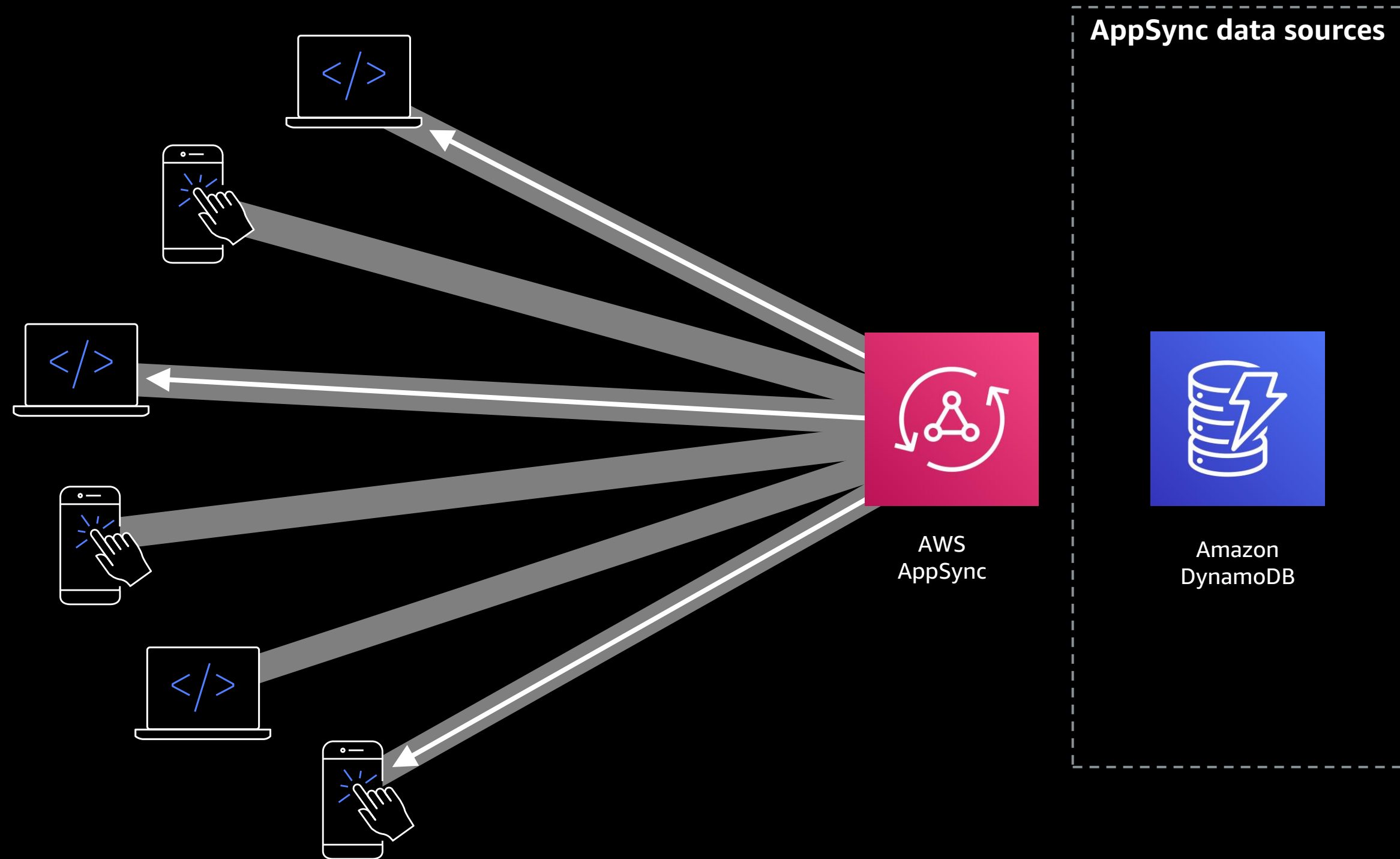
Real-time GraphQL subscriptions



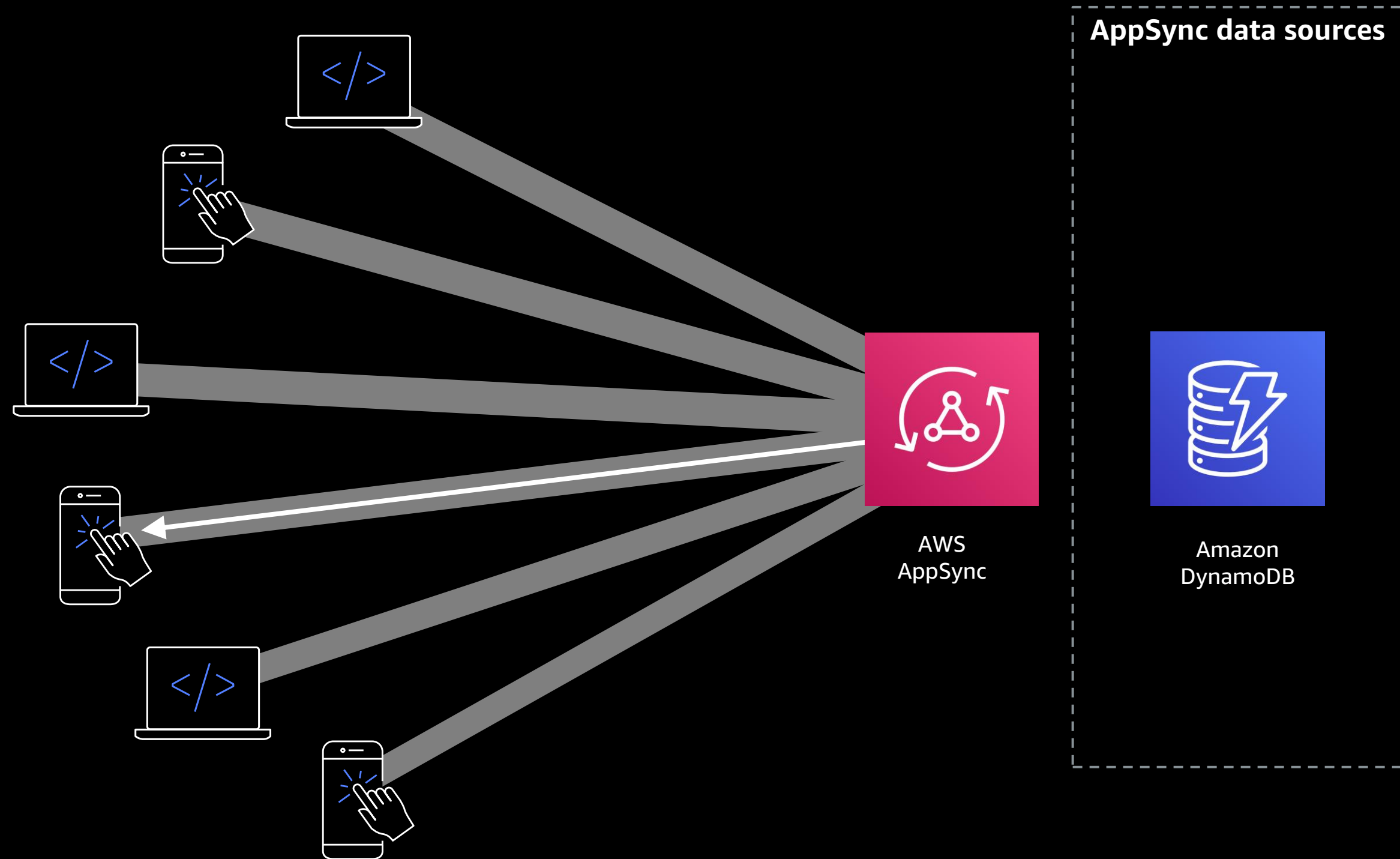
Real-time GraphQL subscriptions



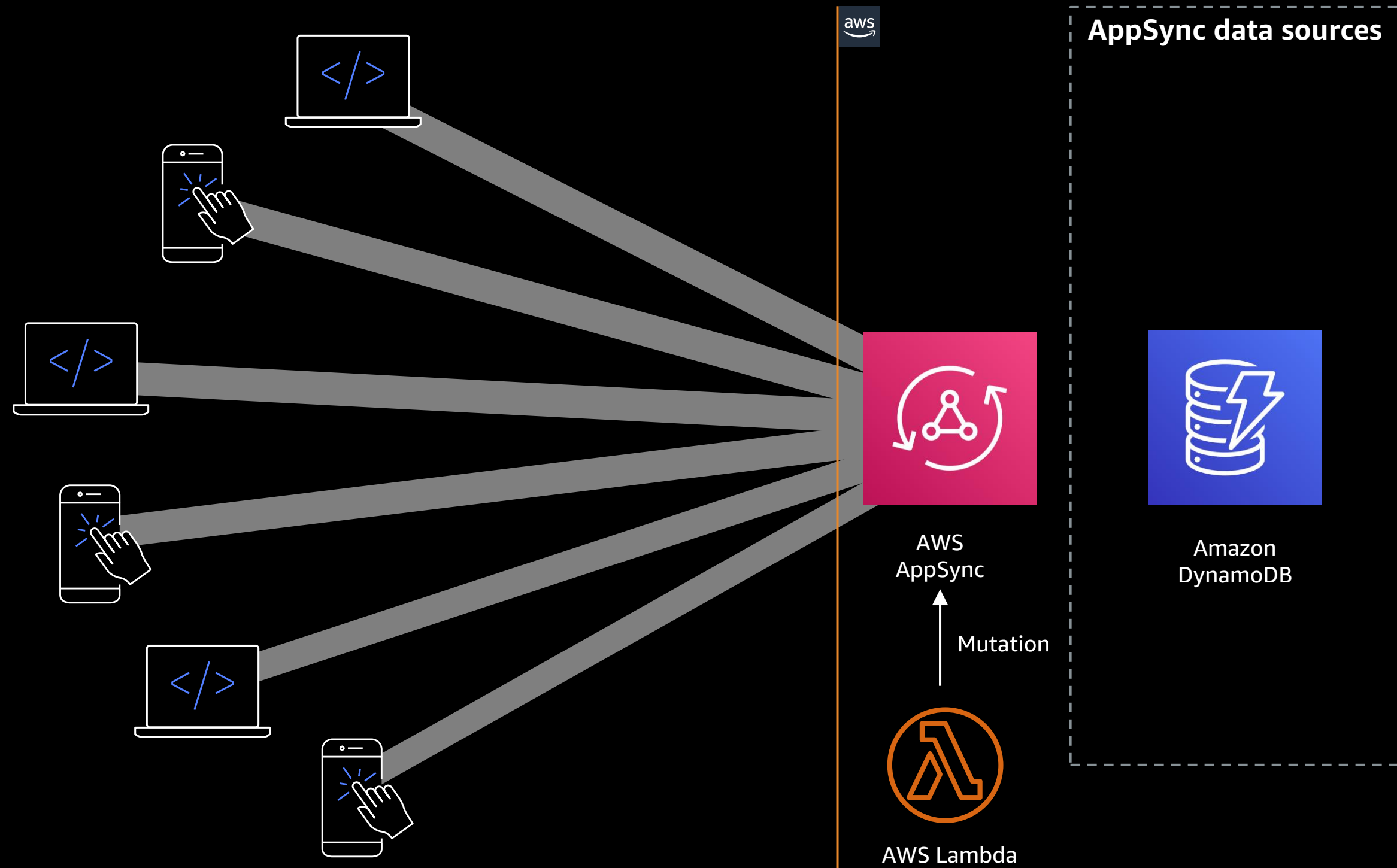
Real-time GraphQL subscriptions



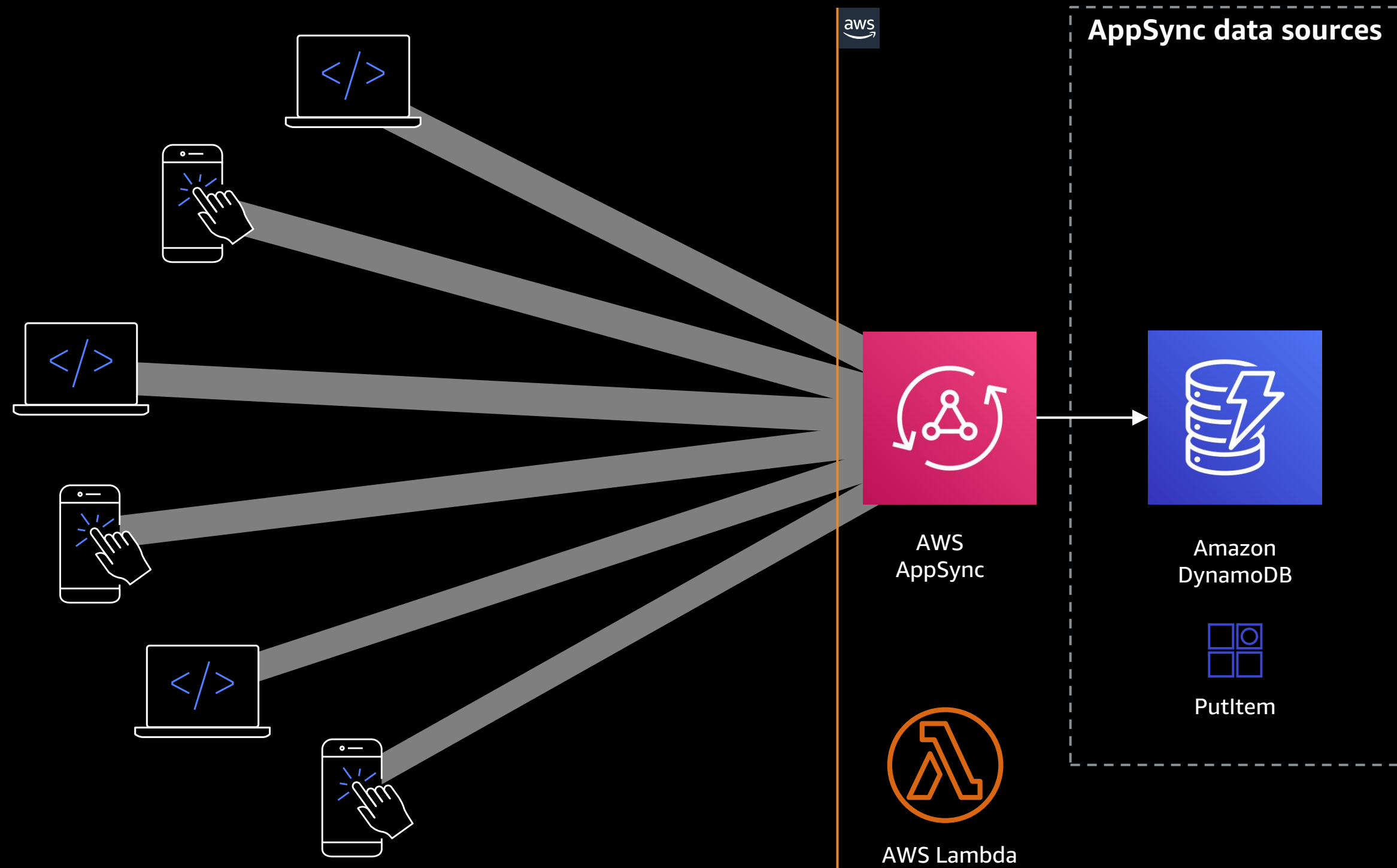
Real-time GraphQL subscriptions



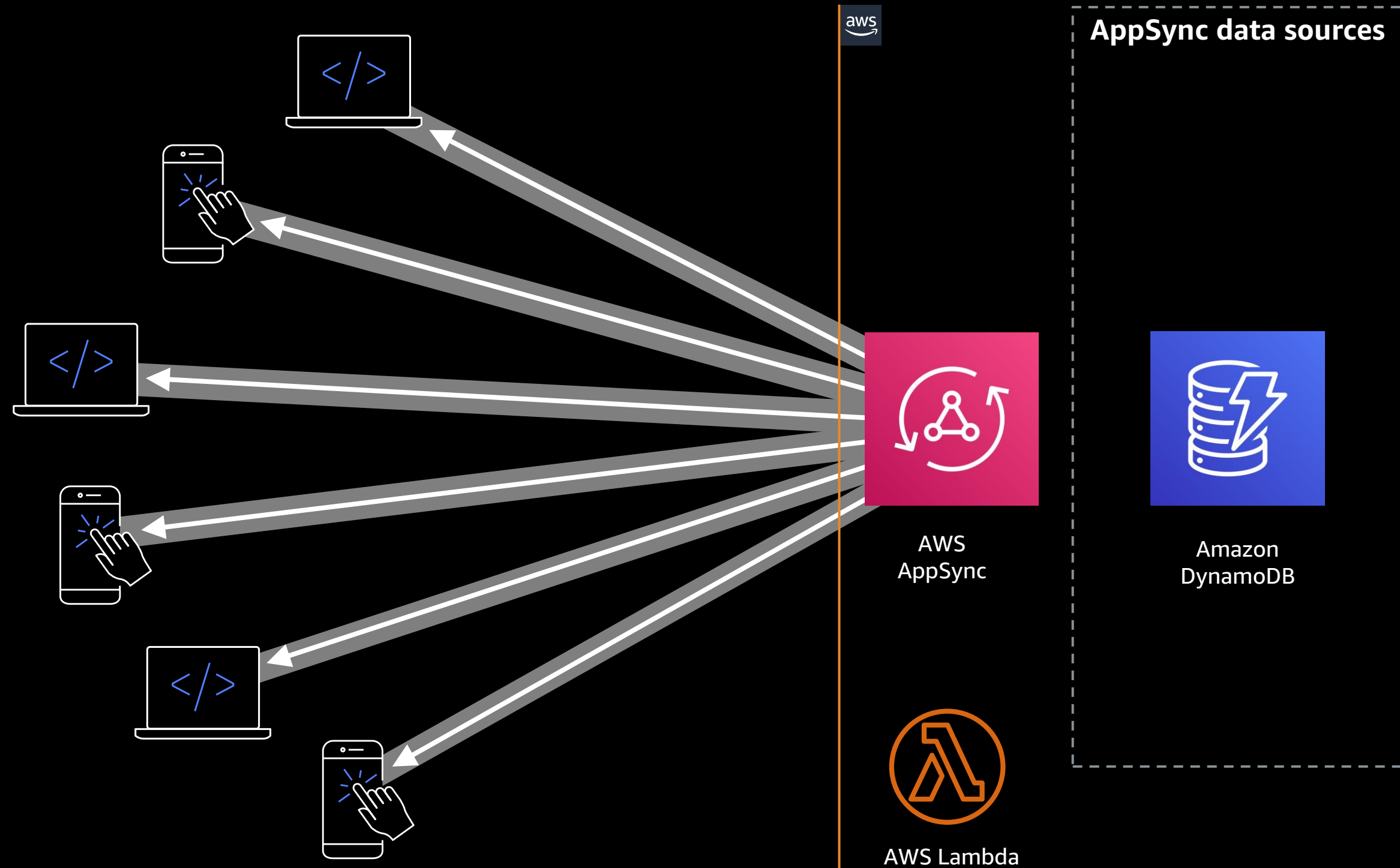
Real-time GraphQL subscriptions



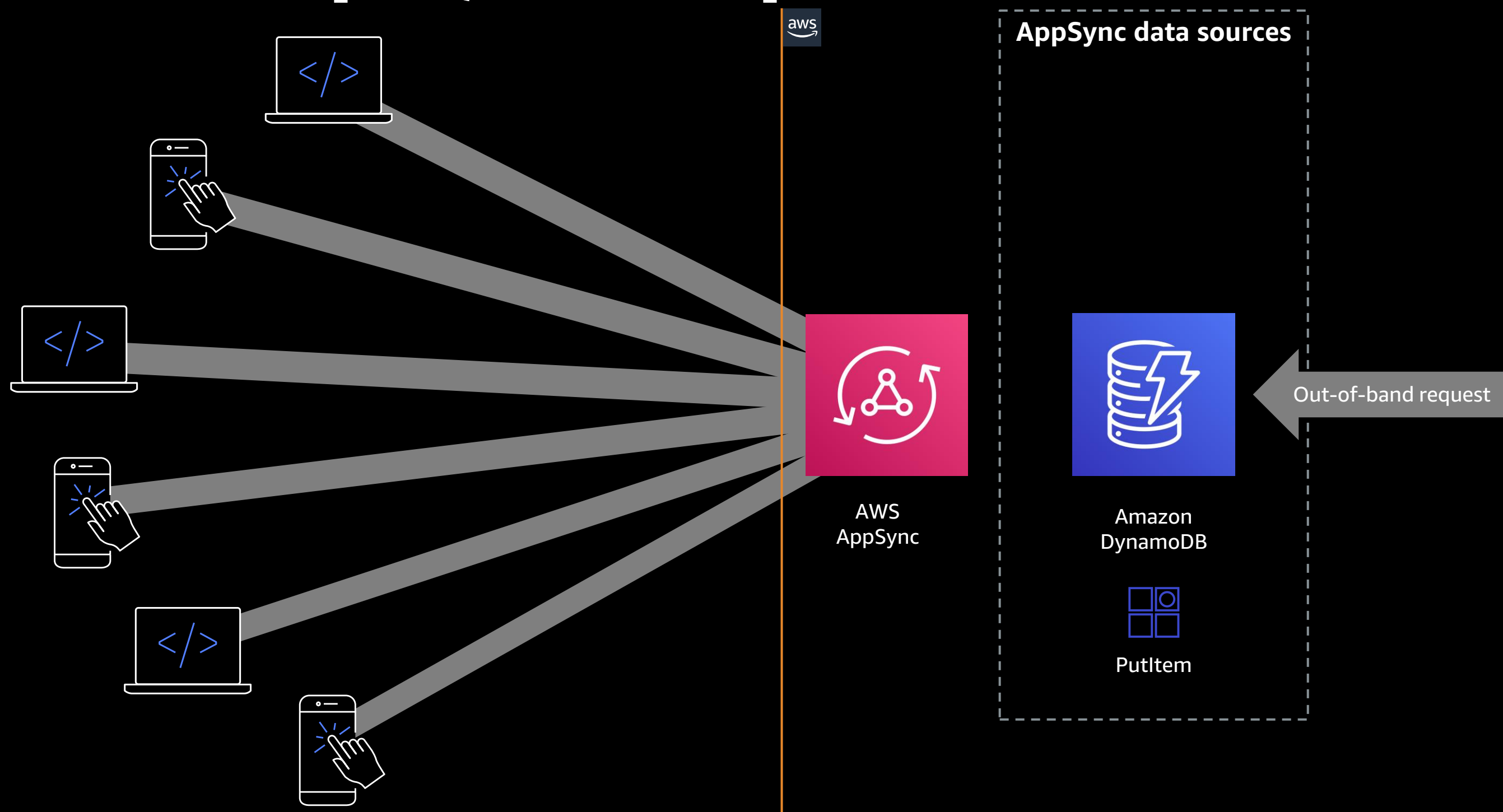
Real-time GraphQL subscriptions



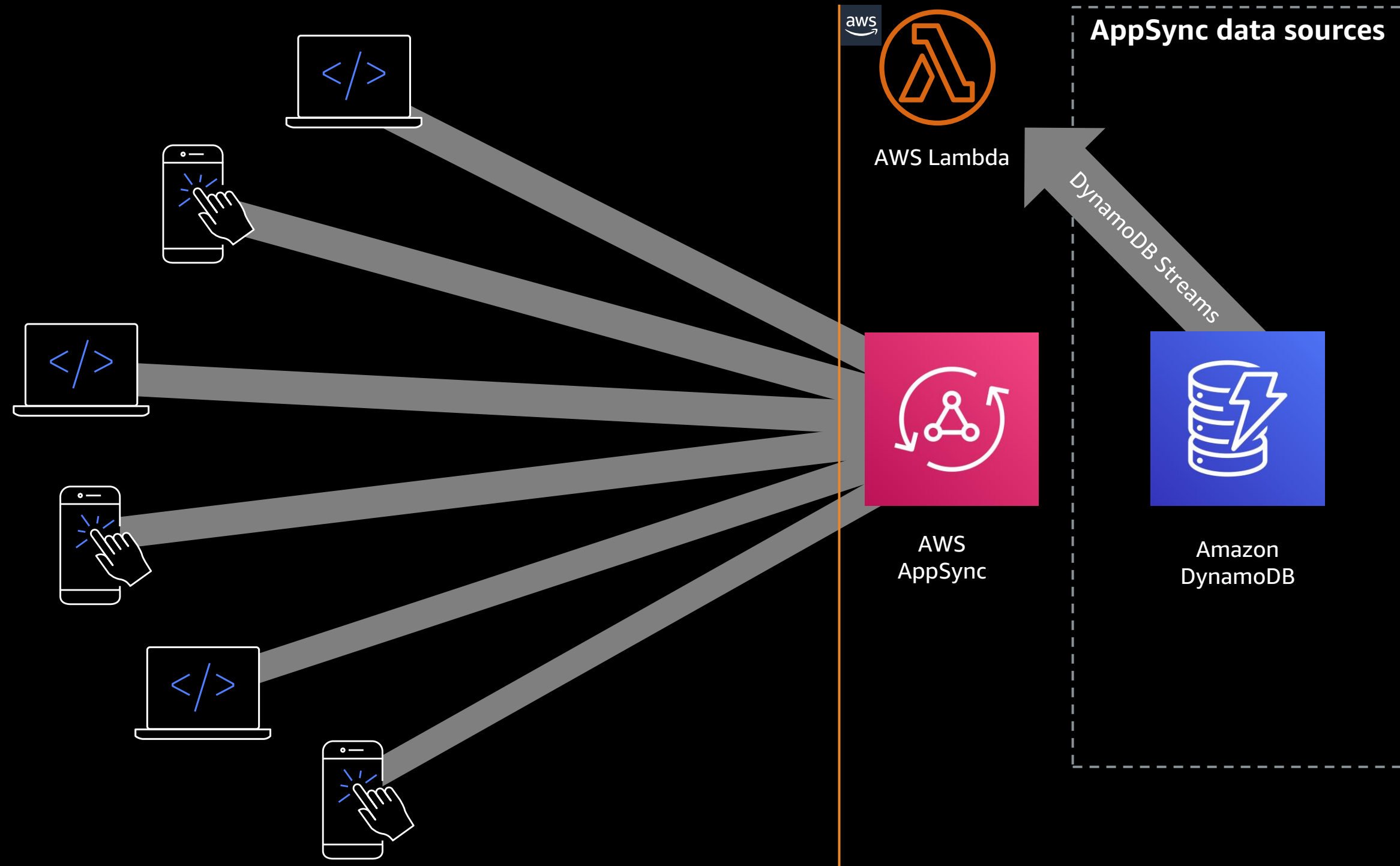
Real-time GraphQL subscriptions



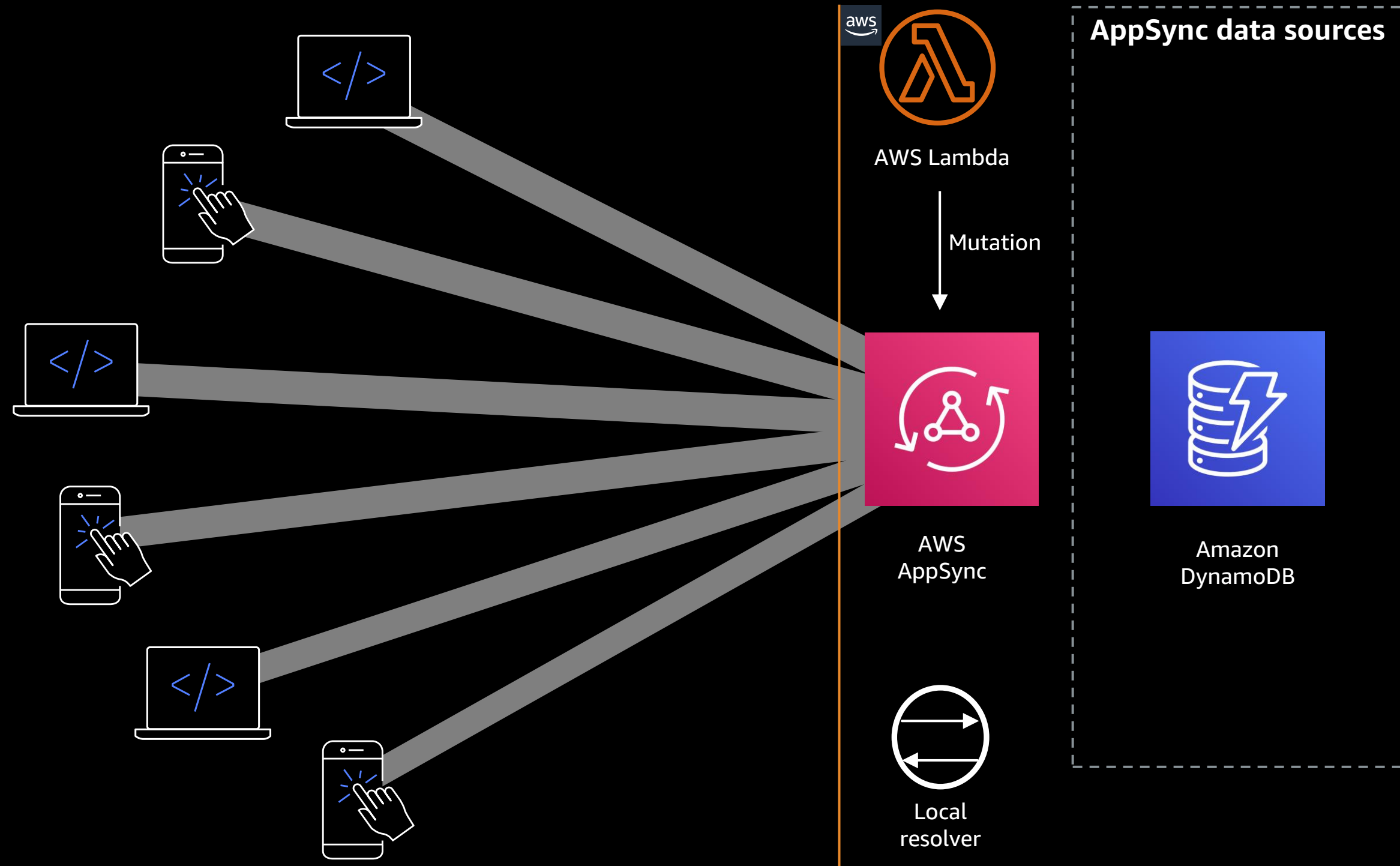
Real-time GraphQL subscriptions



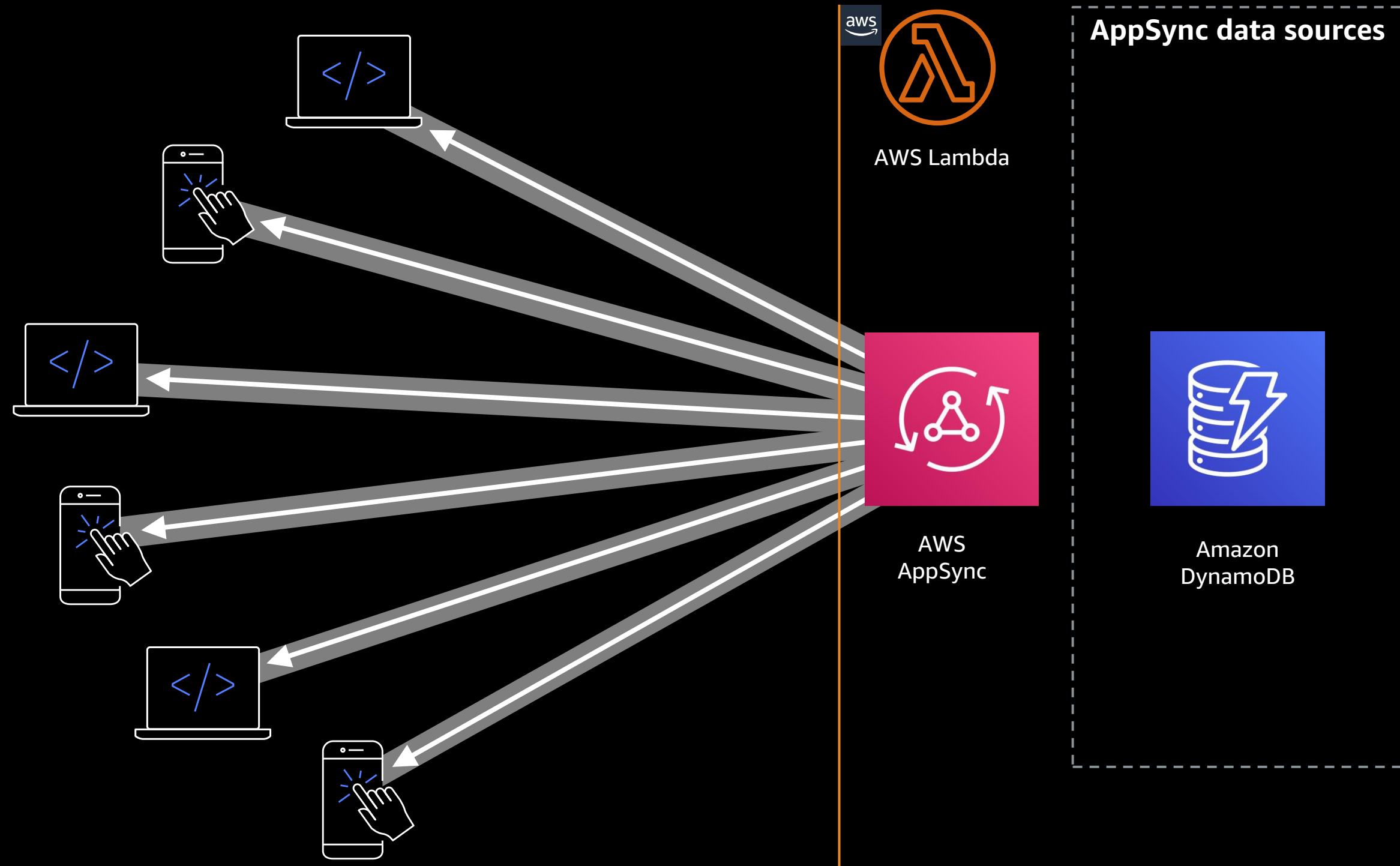
Real-time GraphQL subscriptions



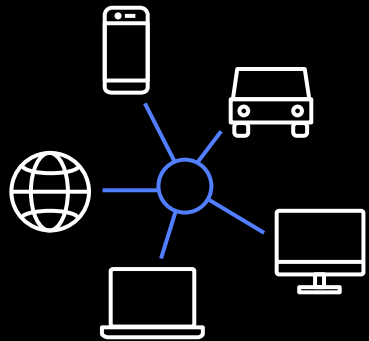
Real-time GraphQL subscriptions



Real-time GraphQL subscriptions



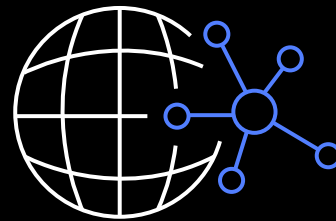
AWS AppSync real-time WebSockets



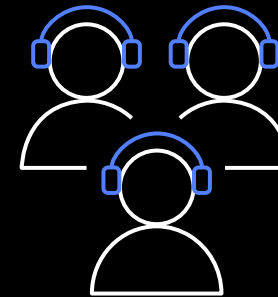
Connection
management



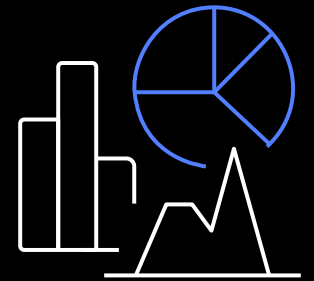
Scalability



Fan-out



Broadcasting



Metrics

Offline and sync

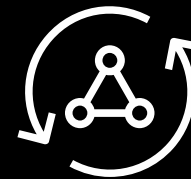


AWS Amplify DataStore



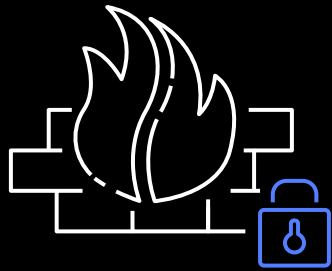
- Local store
- Amazon DynamoDB data sources
- Abstracts GraphQL on the client side
- JavaScript, React Native, iOS, Android
- Local storage (web browser) and SQLite on native platforms

AWS AppSync SDKs



- Local cache
- Any supported AppSync data source
- GraphQL programming model
- JavaScript, React Native, iOS, Android
- Local storage (web browser) and SQLite on native platforms

Managed built-in integrations



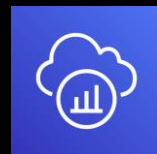
Protection



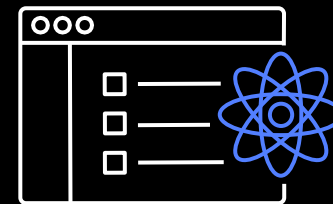
AWS WAF



Observability



AWS X-Ray



Insights



Amazon CloudWatch

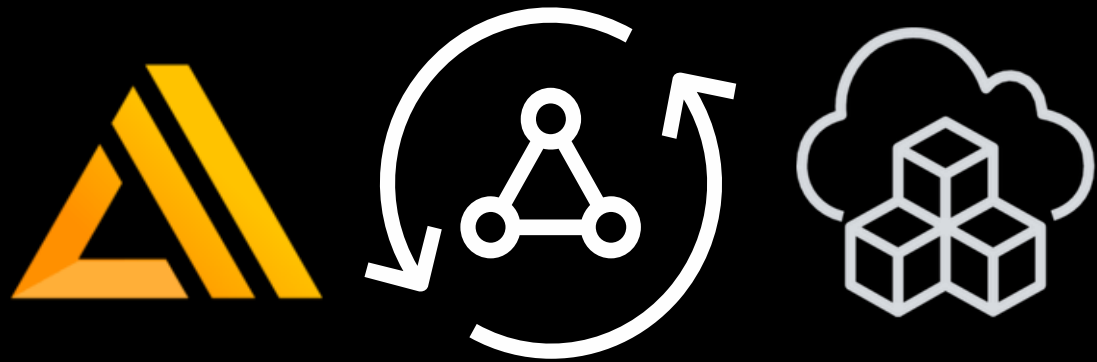


Caching





Amazon ElastiCache

Developer tools



AWS Amplify

DEVELOP

-  Authentication
-  DataStore
-  Storage
-  API
-  Analytics
-  PubSub
-  Predictions
-  Interactions
-  Notifications



Libraries



CLI



Admin UI

NEW!

DELIVER

Static Web Hosting

Fully-managed
Full-stack deployments
CI/CD built in
Pull request previews

Tools



CLI



Console

MANAGE

NEW!

Authentication

Manage users & groups

DataStore

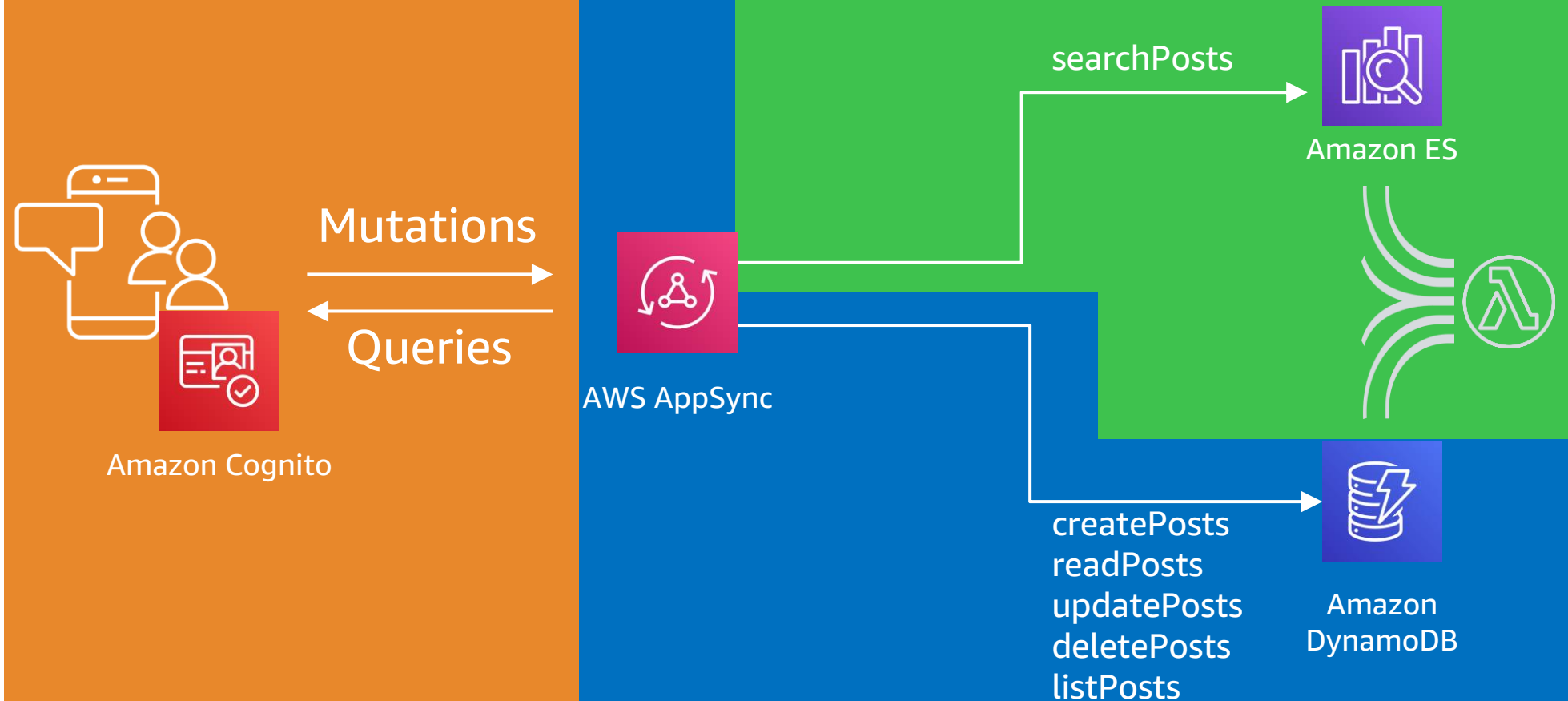
Manage data content



Admin UI

AWS Amplify CLI: GraphQL Transform

```
type Post
@model
@auth(rules: [{allow: owner}])
@searchable
{
  id: ID!
  content: String
  description: String
  ups: Int
  downs: Int
}
```



AWS CDK

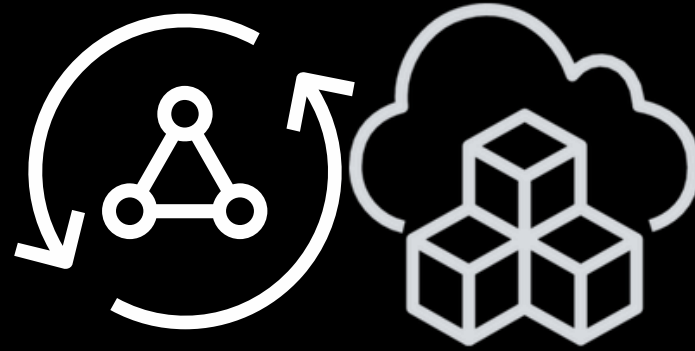
AWS AppSync Construct Library: @aws-cdk/aws-appsync



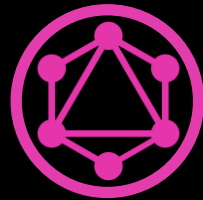
```
1 import * as appsync from '@aws-cdk/aws-appsync';
2 import * as db from '@aws-cdk/aws-dynamodb';
3
4 const api = new appsync.GraphQLApi(stack, 'Api', {
5   name: 'demo',
6   schema: appsync.Schema.fromAsset(join(__dirname, 'schema.graphql')),
7   authorizationConfig: {
8     defaultAuthorization: {
9       authorizationType: appsync.AuthorizationType.IAM
10     },
11   },
12   xrayEnabled: true,
13 });
14
15 const demoTable = new db.Table(stack, 'DemoTable', {
16   partitionKey: {
17     name: 'id',
18     type: db.AttributeType.STRING,
19   },
20 });
21
22 const demoDS = api.addDynamoDbDataSource('demoDataSource', demoTable);
23
24 // Resolver for the Query "getDemos" that scans the DyanmoDb table and returns the entire list.
25 demoDS.createResolver({
26   typeName: 'Query',
27   fieldName: 'getDemos',
28   requestMappingTemplate: MappingTemplate.dynamoDbScanTable(),
29   responseMappingTemplate: MappingTemplate.dynamoDbResultList(),
30 });
31
32 // Resolver for the Mutation "addDemo" that puts the item into the DynamoDb table.
33 demoDS.createResolver({
34   typeName: 'Mutation',
35   fieldName: 'addDemo',
36   requestMappingTemplate: MappingTemplate.dynamoDbPutItem(PrimaryKey.partition('id').auto(), Values.projecting('demo')),
37   responseMappingTemplate: MappingTemplate.dynamoDbResultItem(),
38 });
```



GraphQL development with AWS CDK

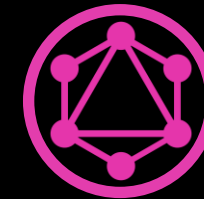


Schema-first

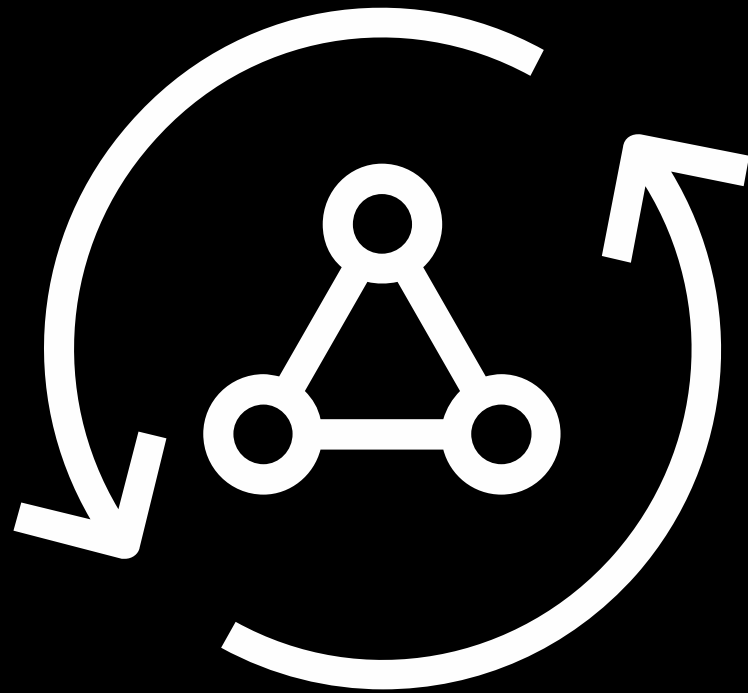


- Schema defined in SDL as your source of truth and forces your code to follow the definitions stored in your schema
- The schema definition must be constantly synced with the resolvers, otherwise it might cause problems
- It's more abstract and less dependent by following the dependency inversion principle

Code-first



- Default option in the CDK
- Schema is defined and implemented programmatically
- The design process begins with coding the resolvers, and the SDL version of the GraphQL schema is a generated artifact
- Modularity, reusability, consistency



- Start effortlessly
- Scale with your business
- Real-time and offline
- Unify and secure access to your distributed data and services
- AWS Amplify integrations: DataStore, GraphQL Transform, Local Mocking, and CodeGen
- Pick your poison: schema-first or code-first with the AWS CDK

AWS AppSync resources

Website

aws.amazon.com/appsync

Docs

docs.aws.amazon.com/appsync

Github

github.com/aws/aws-appsync-community

Blog

aws.amazon.com/appsync/blog/

More resources

aws.amazon.com/appsync/resources/

Live Q&A on Twitch

Join the Amplify and AppSync Team Live!

Weds, Dec. 9: 1-2pm PST / 4-5pm EST

Thurs, Dec.10: 3-4pm PST / 6-7pm EST

Weds, Dec.16: 1-2pm PST / 4-5pm EST

Friday, Dec. 18: 1-2pm PST / 4-5pm EST


Watch live: www.twitch.tv/aws



Thank you!

Ed Lima

Senior Product Manager, AWS

 @ednergizer



Please complete
the session survey