A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light green color. They are positioned diagonally, with the blue one in front of the green one.

# Apollo Federation with Subscription Service



# Problem Space ?

What are we trying to solve with Apollo Federation ?

- Smaller connected microservices vs. large monolithic services.
- Consuming developers able to use a familiar graphql query language without trying to piece together where the data lives
- Need for real-time subscription / web socket support

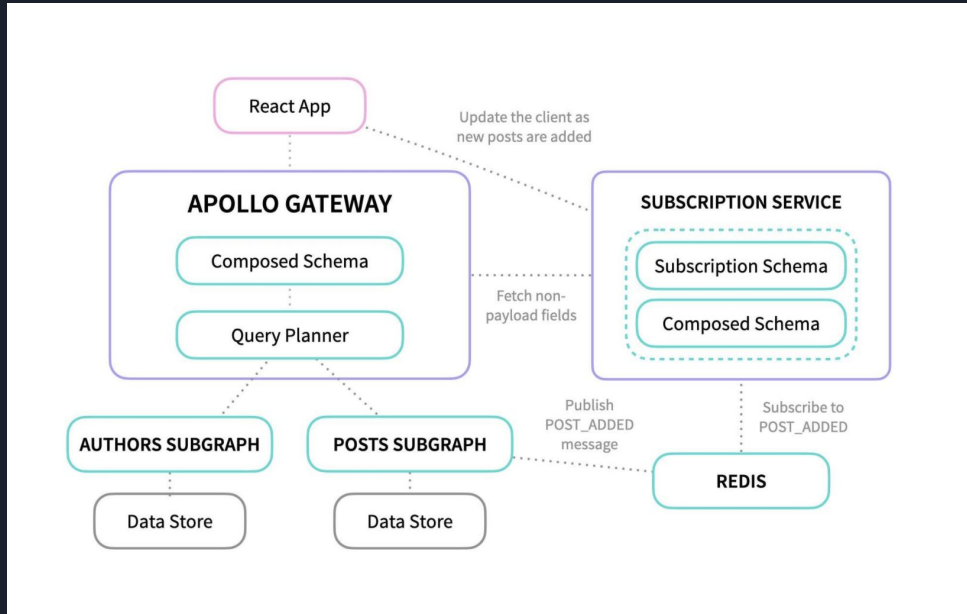


# Players in this demo

- User Service - users are stored in their own database, and have their own endpoint/resolvers.
- Post Service - posts are stored in their own database, yet need to know who created the post.
- Gateway Service - the gateway provides a single endpoint for the consumers - this is the federation service
- Subscription Service - apollo federation does not support subscriptions, but we want to have push notifications to the UI. Redis is the glue.
- ReactApp - consumer of the above services

In an environment like Techscout - posts could represent the Techscout API, or a Techplan API, etc... yet they all need access to the users of the system

# Architecture Diagram



Reference: <https://www.apollographql.com/blog/backend/federation/using-subscriptions-with-your-federated-data-graph/>



# How to run the demo

The entire demo can be run with the docker-compose file. A make file is included. The databases used are just sqlite to reduce more database instantiation.

To run with make just run 'make build' and then 'make up'

Alternatively you can start each piece individually. Start order is:

1. Post service
2. User service
3. Gateway service
4. Subscription service
5. React App

Each service can be

y and typing 'npm start'

A default user is rec

the user-service has been setup to seed an initial user.

```
mutation generateUser{
  createUser(input:{
    name:"Steve Gentile"
  }){
    id
    name
  }
}
```

Additional commands can be found in the readme file.