Apollo: Performance Management Playbook

LEARNING APOLLO CLIENT CACHING TECHNIQUES

Peter Kellner
DEVELOPER, CONSULTANT AND AUTHOR

@pkellner linkedin.com/in/peterkellner99 ReactAtScale.com

Why Use Cache in Apollo GraphQL Client

Good for server

Fewer roundtrips from client to server

Less computational load required

Less network traffic

Lower cost

Good for user

User interface more responsive

Works better on slower connections

Enables optimistic UI

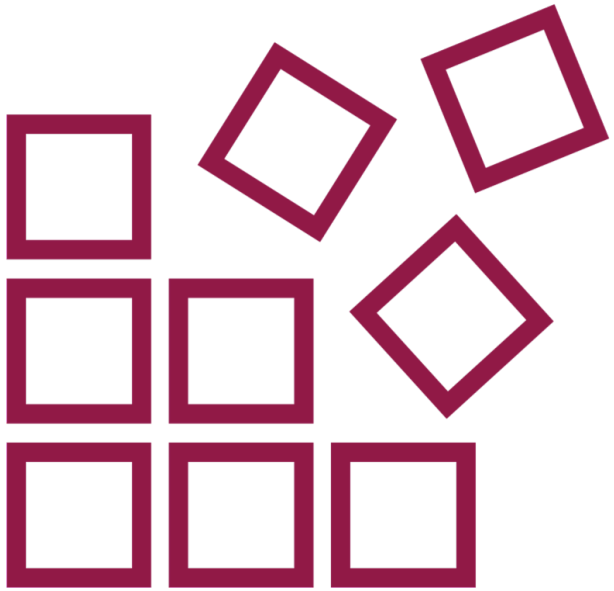
Leverage local compute power

```
const apolloClient = new ApolloClient({
  uri: "http://localhost:4000",
  cache: new InMemoryCache(),
});
```

Apollo Client Configuration

What's Coming in This Module

- Normalizing data for caching
- Reading and writing cache data
- Enhancing insert and delete mutations
- Implementing an optimistic UI



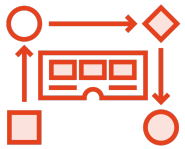
We will be using the GraphQL server built in module 2

We will be extending the React app we build with Next.js and Apollo in module 3

The Plan for Understanding Apollo Cache Normalization

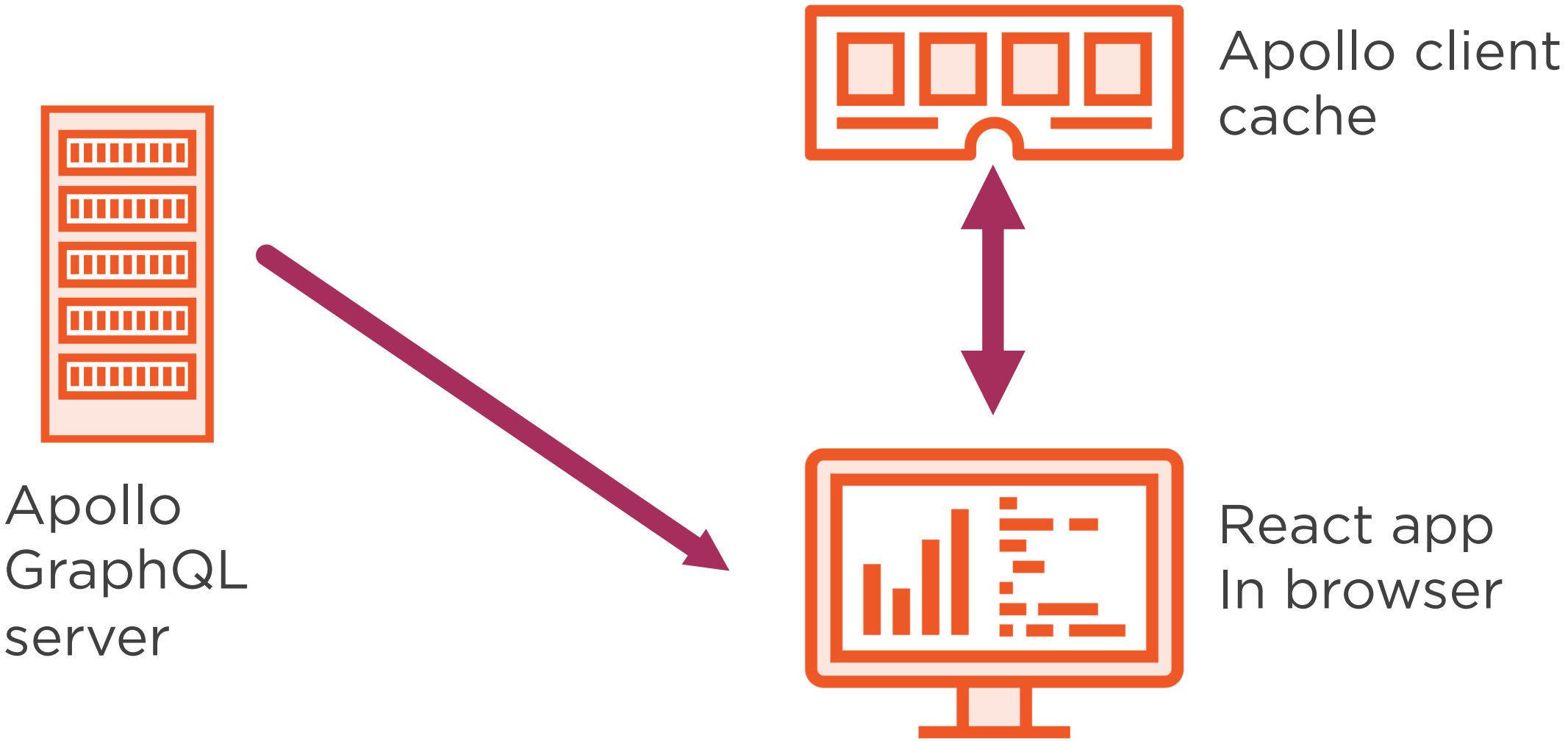
Look at Query sent to Apollo Server

Look at results returned from Apollo Server



Look at Apollo Cache created from processing the query

Apollo Cache Interactions



Implementing an Optimistic UI in Apollo client is easy

We will implement Optimistic UI for our speaker favorite icon and the trash icon

Takeaways

- Apollo cache tied to React state
- Fixed insert and delete mutations to be cache aware
- Implemented Optimistic UI pattern with just declarative code
- Implement UI data sorting by sorting Apollo cache items in memory