🖼️ Session 3: Frontend with GraphQL

Tech Stack

- React + Apollo Client
- OR Vanilla JS + Fetch + GraphQL

Goals:

- Display posts from the backend
- Add new posts from the UI

🔌 Apollo Client Setup (React)

```
import { ApolloClient, InMemoryCache, ApolloProvider } from '@apollo/client'

const client = new ApolloClient({
  uri: 'http://localhost:4000/',
  cache: new InMemoryCache()
})
```

🔗 https://www.apollographql.com/docs/react

🧪 Example Query Component

```
import { useQuery, gql } from '@apollo/client'

const GET_POSTS = gql`
  query {
    posts {
      id
      title
    }
  }
`


function Posts() {
  const { loading, error, data } = useQuery(GET_POSTS)
  if (loading) return <p>Loading...</p>
  if (error) return <p>Error: {error.message}</p>

  return (
    <ul>{data.posts.map(p => <li key={p.id}>{p.title}</li>)}</ul>
  )
}
```

🧼 Alternative: Fetch + GraphQL (Vanilla JS)

```
fetch('http://localhost:4000/', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ query: '{ posts { id title } }' })
})
.then(res => res.json())
.then(data => console.log(data))
```

🔚 Conclusion

✅ You now know:

- What GraphQL is and how it's different from REST
- How to define and query a schema
- How to build a backend with GraphQL
- How to consume GraphQL from the frontend

🚀 Keep building!

📎 Resources

- https://graphql.org/learn
- https://www.apollographql.com/docs/
- https://graphqlweekly.com
- https://hasura.io/learn

Thanks for joining! 🙌

Leave feedback on: https://mlh.link/ghwfeedback