
# TDDC73
# Lecture 3

# Agenda

- Questions
- Labs
- That model thing (cont. from the last lecture)
- Screen navigation
- Network
  - REST and/or GraphQL


LiU EXPANDING REALITY

# Labs

- Webreg
- Flutter and React-Native
  - It is ok to develop and test in the browser
  - nvm for React-Native (recommended by me)
- Lab 3 “Trending”
  - Pick an attribute to sort on
    - Forks
    - Issues
    - Find one you think is funny/relevant/Interesting

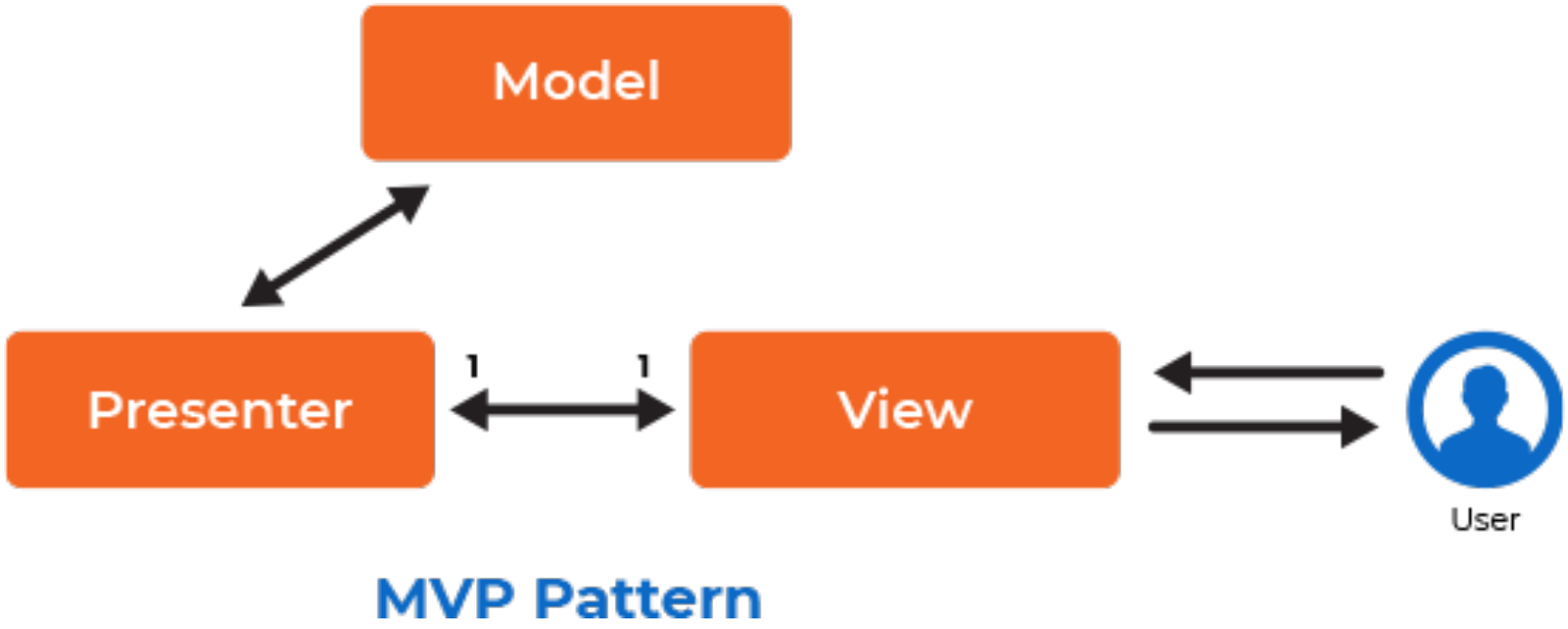

LiU EXPANDING REALITY

# Organisation of code

- Biggest challenge of UI development (Ok one of )
  - Maintaining  the correct and same state between the UI and the system/model/backend
- Separation of concerns
- Architecture Presentation Patterns
  - MVC
  - MVP
  - MVVM


LiU EXPANDING REALITY

# Model-View-Presenter MVP

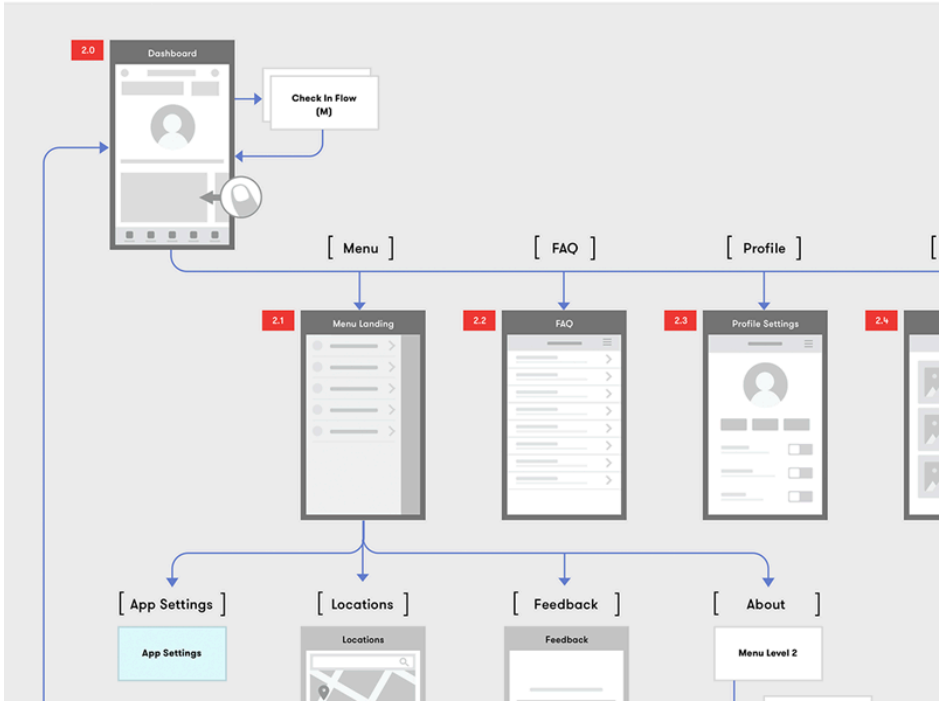


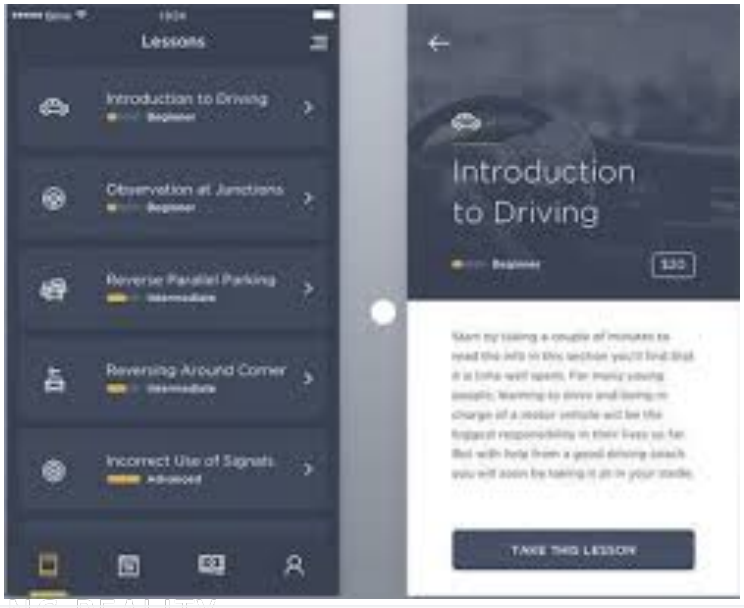
# Demo

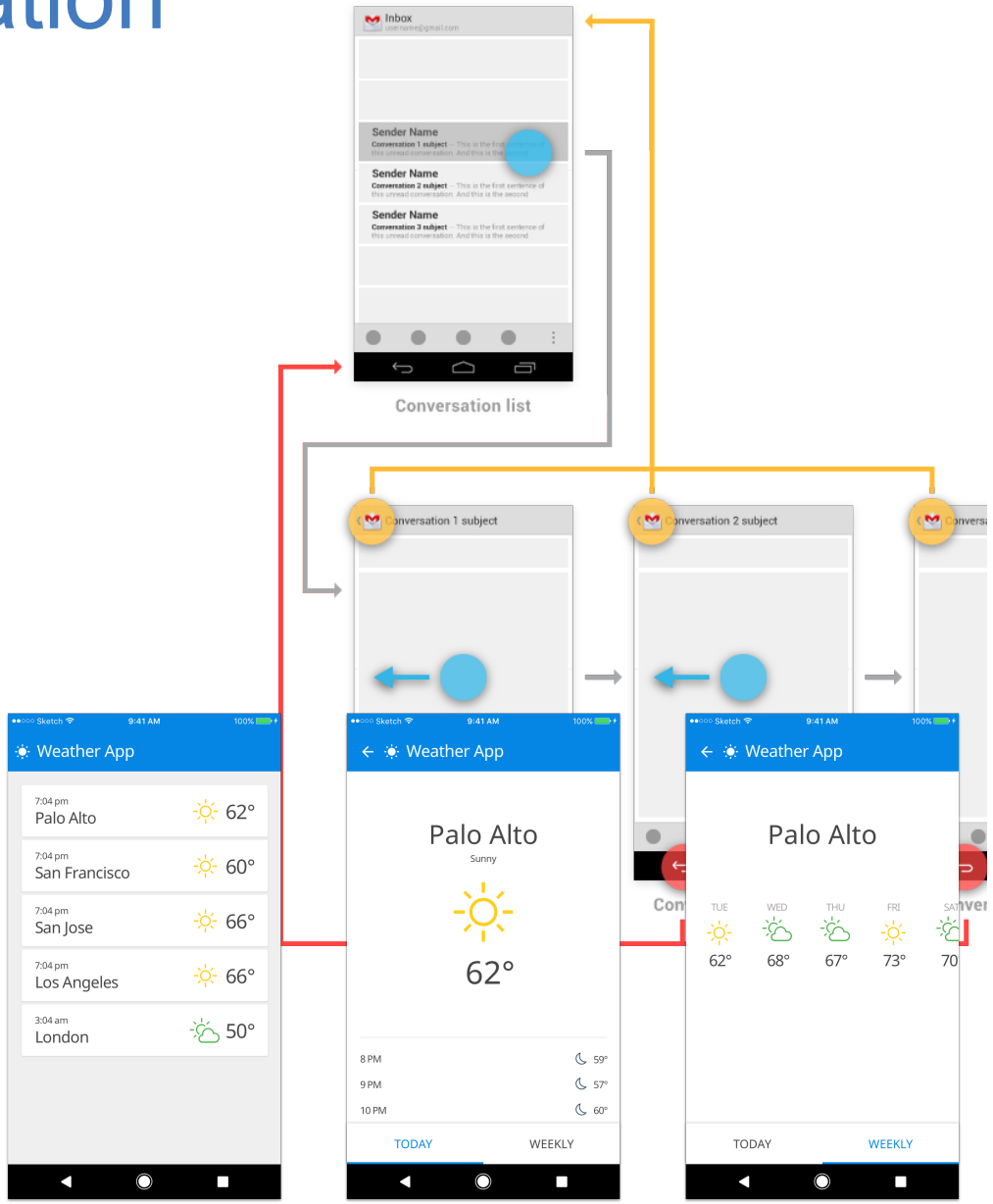
- Continue the Android Adapter example


LiU EXPANDING REALITY

# Navigation


2.0
Dashboard
Check In Flow
(M)
[ Menu ]
[ FAQ ]
[ Profile ]
2.1
Menu Landing
2.2
FAQ
2.3
Profile Settings
2.4
[ App Settings ]
[ Locations ]
[ Feedback ]
[ About ]
App Settings
Locations
Feedback
Menu Level 2


Lessons
Introduction to Driving
Observation at Junctions
Reverse Parallel Parking
Reversing Around Corner
Incorrect Use of Signals
Introduction
to Driving
TAKE THIS LESSON


Inbox
Sender Name
Sender Name
Sender Name
Conversation list
Conversation 1 subject
Conversation 2 subject
Weather App
7:04 pm
Palo Alto
62°
7:04 pm
San Francisco
60°
7:04 pm
San Jose
66°
7:04 pm
Los Angeles
66°
3:04 am
London
50°
Palo Alto
Sunny
62°
8 PM
59°
9 PM
57°
10 PM
60°
TODAY
WEEKLY
Palo Alto
TUE
WED
THU
FRI
62°
68°
67°
73°
TODAY
WEEKLY

# Flutter

- Routes
  - Pages/Screens
- Navigation
  - To move between Routes
  - Push and Pop
- Hero
  - Animation between Routes


LiU EXPANDING REALITY

# Routes

```
class FirstRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('First Route'),
      ),
      body: Center(
        child: RaisedButton(
          child: Text('Open route'),
         },
        ),
      ),
    );
  }
}
```

```
class SecondRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Second Route"),
      ),
      body: Center(
        child: RaisedButton(¨
         child: Text('Go back!'),
        ),
      ),
    );
  }
}
```


LiU EXPANDING REALITY

# Navigation

```
child: RaisedButton(
          child: Text('Open route'),
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(builder:
(context) => SecondRoute()),
            );
          },
        ),
```

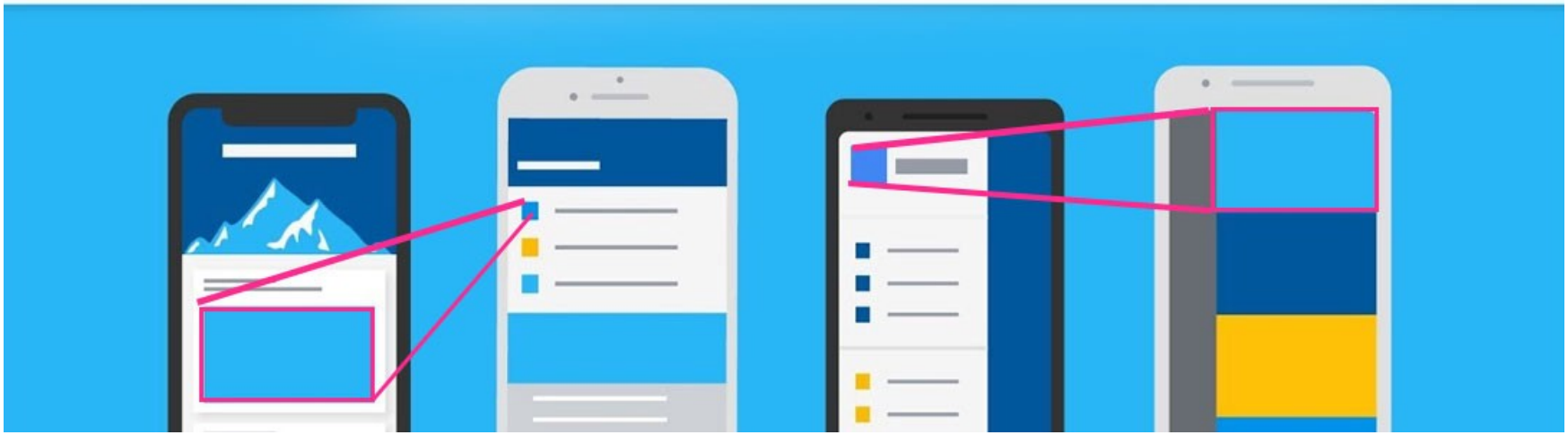
```
child: RaisedButton(
          onPressed: () {
            Navigator.pop(context);
          },
          child: Text('Go back!'),
        ),
```

```
routes: {
      '/': (context) => FirstScreen(),
      '/info': (context) => SecondRoute(),
    },
```

```
Navigator.pushNamed(context, '/info');
```


LiU EXPANDING REALITY

# Hero - transitions with animation
# Helps users through animation



```
Hero(
  tag: 'imageHero',
  child: Image.network(
    'https://picsum.photos/250?image=9',
  ),
);
```


LiU EXPANDING REALITY

# React-Native

- React-Native
  - Multiple ways, both standard and external libraries
- Screens
  - Stack-Navigator
  - Routes describe Screens


LiU EXPANDING REALITY

# Screen (Just a component)

```
function AndersScreen({ navigation }) {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
      <Button
        title="Go to TDDC73"
        onPress={() => navigation.navigate('TDDC73')}
      />
    </View>
  );
}
```

```
Function App(){
Yada yada ….

return (<NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Anders" component={AndersScreen} />
        <Stack.Screen name="TDDC73" component={TDDC73Screen} />
      </Stack.Navigator>
    </NavigationContainer>);
}
```

# Push,navigate,Back

```
function TDDC73Screen({ navigation }) {
  return (
      ……
       <Button
         title="Go to Anders... again"
         onPress={() => navigation.navigate('Anders')}
       />
       <Button
         title="Go back"
         onPress={() => navigation.goBack()}
       />
        <Button
         title="Go to Anders"
         onPress={() => navigation.push('Anders')}
       />
    </View>
  );
}
```


LiU EXPANDING REALITY

# Standard Android

- Basic
  - Activities
  - Fragments
- Navigator (like flutter and react) in googles Architecture pattern/ solution


LiU EXPANDING REALITY

# Network

- REST



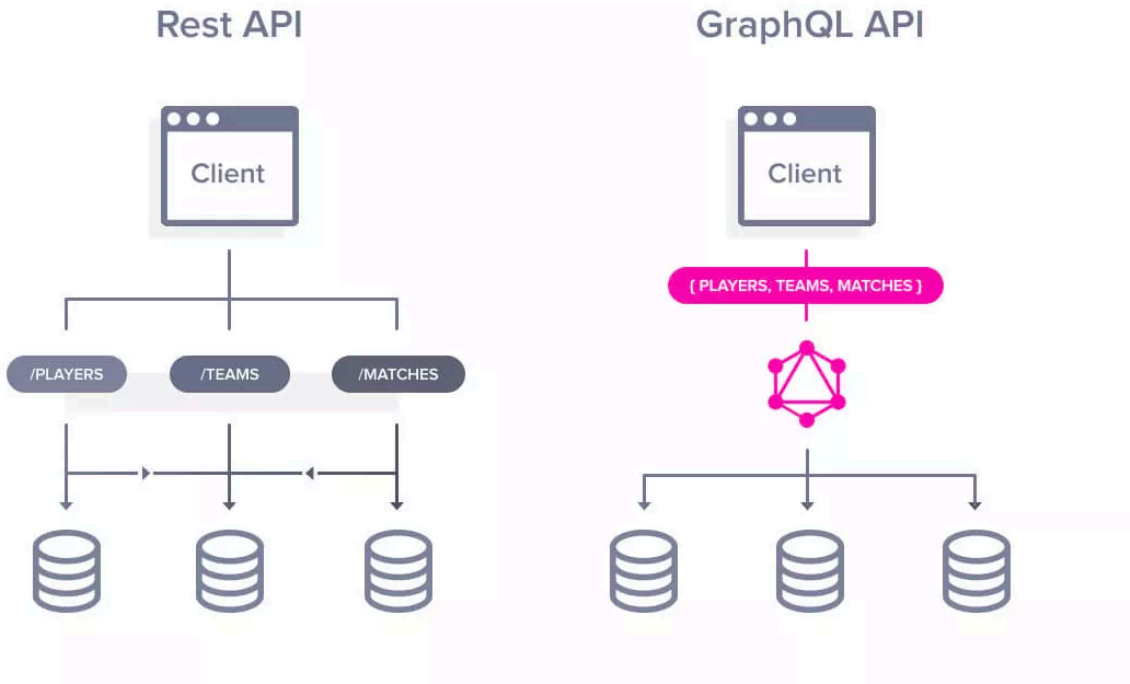

LiU EXPANDING REALITY

# DEMO old school


LiU EXPANDING REALITY

# Network

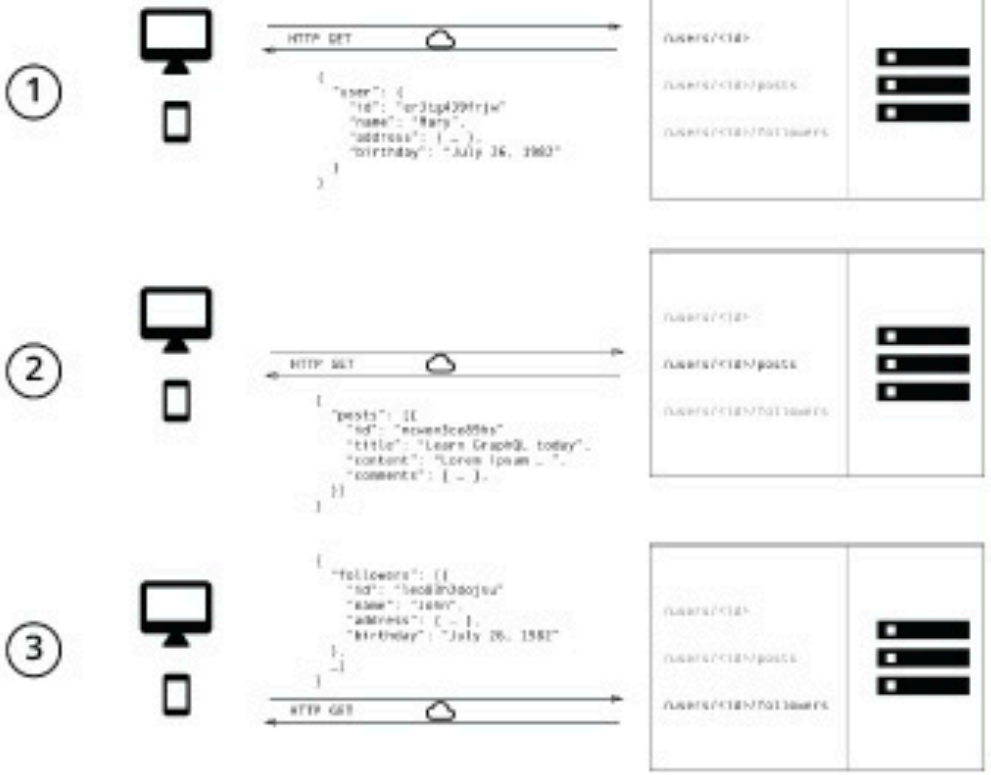
- GraphQL
  - Putting REST to rest (sorry for that one)



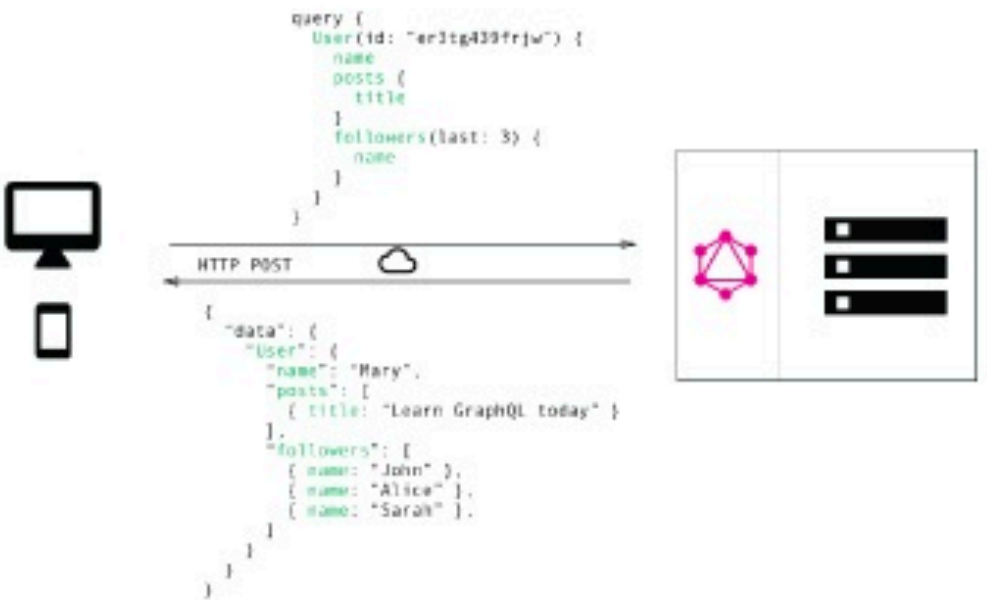


LiU EXPANDING REALITY

# REST API




# GraphQL



# REST vs. GraphQL 🍕

## REST

```
GET /api/pizza/cheese
GET /api/pizza/supreme
GET /api/pizza/margherita
```

- Filter down the data
- Perform waterfall requests for related data
- Aggregate the data yourself

## GraphQL

```
query {
  pizza {
    cheese {
      mozzarella
    }
    toppings {
      basil
      mushrooms
    }
  }
}
```

- Receive exactly what you ask for
- No aggregating or filtering data


LiU EXPANDING REALITY

# GraphQL

- A query language runtime  for  server-side
- Front-end driven
  - You decide what you want
  - Subway vs Pressbyrån
- Singel endpoint
- Typed


LiU EXPANDING REALITY

# Schema
# Filelds and Types

```
type Query {
  me: User
}

type User {
  id: ID
  name: String
}
```

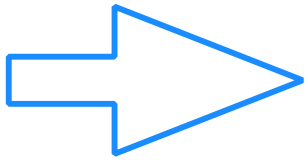
```
type Query {
  bookById(id: ID): Book
}

type Book {
  id: ID
  name: String
  pageCount: Int
  author: Author
}

type Author {
  id: ID
  firstName: String
  lastName: String
}
```

# Queries

```
{
  bookById(id: "book-1"){
    id
    name
    pageCount
    author {
      firstName
      lastName
    }
  }
}
```



```
{
  "bookById":
  {
    "id":"book-1",
    "name":"Harry Potter and the Philosopher's Stone",
    "pageCount":223,
    "author": {
      "firstName":"Joanne",
      "lastName":"Rowling"
    }
  }
```


LiU EXPANDING REALITY

# Apollo-graphql

- A graphql SDK
  - Support for both back-end and front-end
  - Support for multiple languages and platform
  - Including  Android support (kotlin and java)
- In god public-service spirit should we mention that there are many other SDK for GraphQL


LiU EXPANDING REALITY

# Working with graphQL (Apollo-graphQL in particular )

- Install required libraries (use the build tools if possible)
- Download the schema (schema.json)
- Write your queries
- Execute your queries
- Work with the result


LiU EXPANDING REALITY

# Writing the query

## LaunchDetails.graphql

```
query LaunchDetails($id:ID!) {
  launch(id: $id) {
    id
    site
    mission {
      name
      missionPatch(size:LARGE)
    }
  }
}
```


LiU EXPANDING REALITY

# Executing the Query

```
ApolloClient apolloClient = ApolloClient.builder()
        .serverUrl("https://your.domain/graphql/endpoint")
        .build();

// Then enqueue your query
apolloClient.query(new LaunchDetailsQuery("83"))
        .enqueue(new ApolloCall.Callback<LaunchDetailsQuery.Data>() {
            @Override
            public void onResponse(@NotNull Response<LaunchDetailsQuery.Data> response)
                Log.e("Apollo", "Launch site: " + response.getData().launch.site);
            }

            @Override
            public void onFailure(@NotNull ApolloException e) {
                Log.e("Apollo", "Error", e);
            }
        });
```


LiU EXPANDING REALITY

# Apollo-GraphQL: React-Native

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

const client = new ApolloClient({
  uri: 'https://48p1r2roz4.sse.codesandbox.io',
  cache: new InMemoryCache()
});

client
  .query({
    query: gql`
      query GetRates {
        rates(currency: "USD") {
          currency
        }
      }
    `
  })
  .then(result => console.log(result));
```


LiU EXPANDING REALITY

# Flutter
# Not Apollo

```
import 'package:graphql_flutter/
graphql_flutter.dart';
ValueNotifier<GraphQLClient> client = ValueNotifier(
    GraphQLClient(
      cache: InMemoryCache(),
      link: ‘https://link.to.endpointe,
    ),
  );
String readRepositories = """
  query ReadRepositories(\$nRepositories: Int!) {
    viewer {
      repositories(last: \$nRepositories) {
        nodes {
          id
          name
          viewerHasStarred
        }
      }
    }
  }
""";
```

```
Query(
  options: QueryOptions(
    documentNode: gql(readRepositories),
   variables: {
      'nRepositories': 50,
    },
    pollInterval: 10,
  ),
  builder: (QueryResult result, { VoidCallback
refetch, FetchMore fetchMore }) {
    if (result.hasException) {
        return Text(result.exception.toString());
    }
    if (result.loading) {
      return Text('Loading');
    }
   List repositories = result.data['viewer']
['repositories']['nodes'];
//do what you need with the result
  },
);
// ...
```

# You pick your poison


LiU EXPANDING REALITY