CONNECT TO INTERNET

Tim Dosen Mobpro 1

D3 Rekayasa Perangkat Lunak Aplikasi
Fakultas Ilmu Terapan

PERMISSIONS ON ANDROID

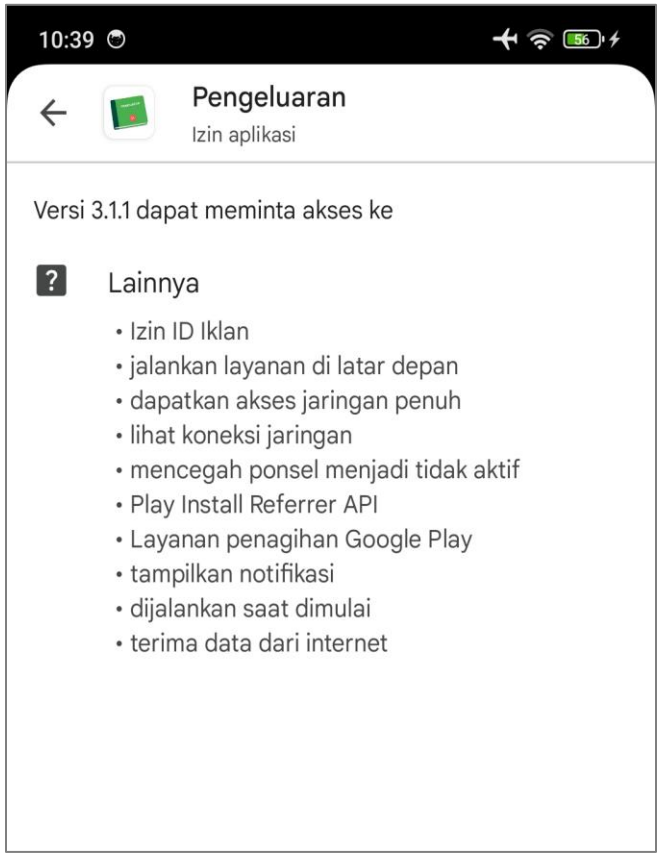
To perform network operations, your app manifest **must** include permission:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
```

App permissions help support **user privacy** by protecting access to:

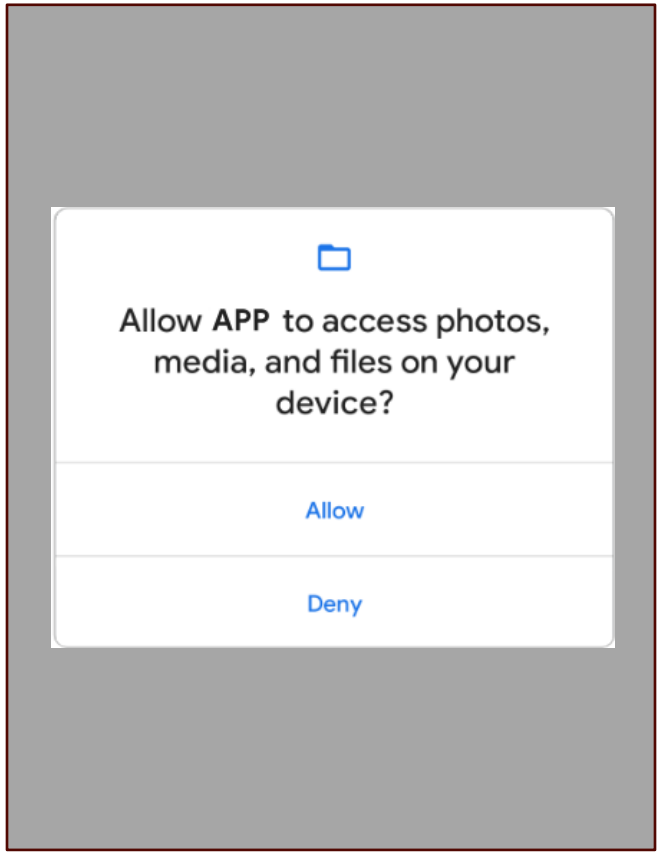
- Restricted data, such as system state and users' contact information
- Restricted actions, such as connecting to a paired device & recording audio

INSTALL-TIME PERMISSIONS



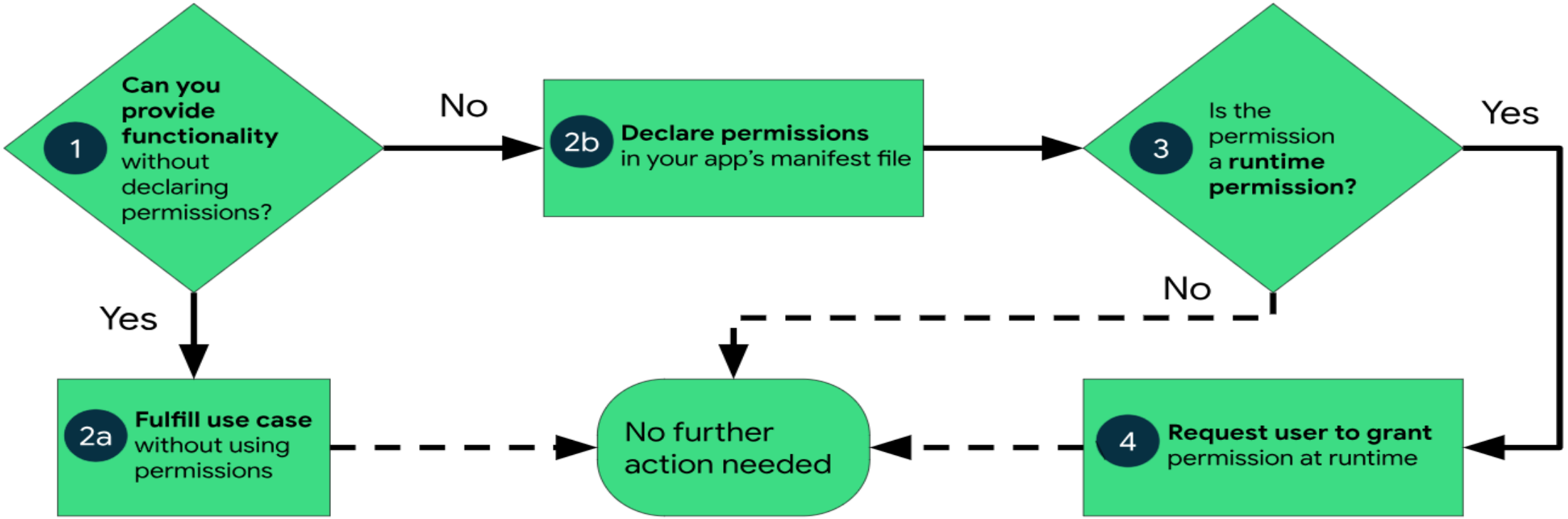
- When you declare install-time permissions, an app store presents an install-time permission notice to the user when they view an app's details page.
- The system **automatically grants** your app the permissions when the user installs your app.
- Both the INTERNET and ACCESS_NETWORK_STATE permissions are normal permissions, which means they're granted at install time.

RUNTIME PERMISSIONS



- Also known as dangerous permissions. Many runtime permissions access private user data, for examples: location and contact information.
- You need to request runtime permissions in your app before you can access the restricted data or perform restricted actions.
- **Don't assume** that these permissions have been previously granted—check them and, if needed, request them before each access.

PERMISSIONS WORKFLOW



PERMISSIONS BEST PRACTICES

- Request a **minimal number** of permissions. Your app should request only the permissions that it needs to complete that action.
- Consider your app's dependencies. **Be aware** of the permissions that each dependency requires and what those permissions are used for.
- Associate runtime permissions with **specific actions**. Request permissions as late into the flow of your app's use cases as possible.
- **If the user denies** or revokes a permission that a feature needs, gracefully degrade your app so that the user can continue using your app, possibly by disabling the feature that requires the permission.

RESTFUL WEB SERVICES

- Most web servers today run web services using a common stateless web architecture **known as REST**, which stands for REpresentational State Transfer.
- Requests are made to RESTful web services in **a standardized way**, via URL that specifies where a resource exists and the mechanism for retrieving it.
- The response from a web service is formatted in **common data formats**, like XML (eXtensible Markup Language) or JSON (JavaScript Object Notation).
- The JSON format represents structured data in **key-value pairs**.

JSON: KEY-VALUE PAIRS

```
{
    "size": 9.5,
    "wide": true,
    "country-of-origin": "usa",
    "style": {
            "categories": [ "boot", "winklepicker" ],
            "color": "black"
    }
}
```

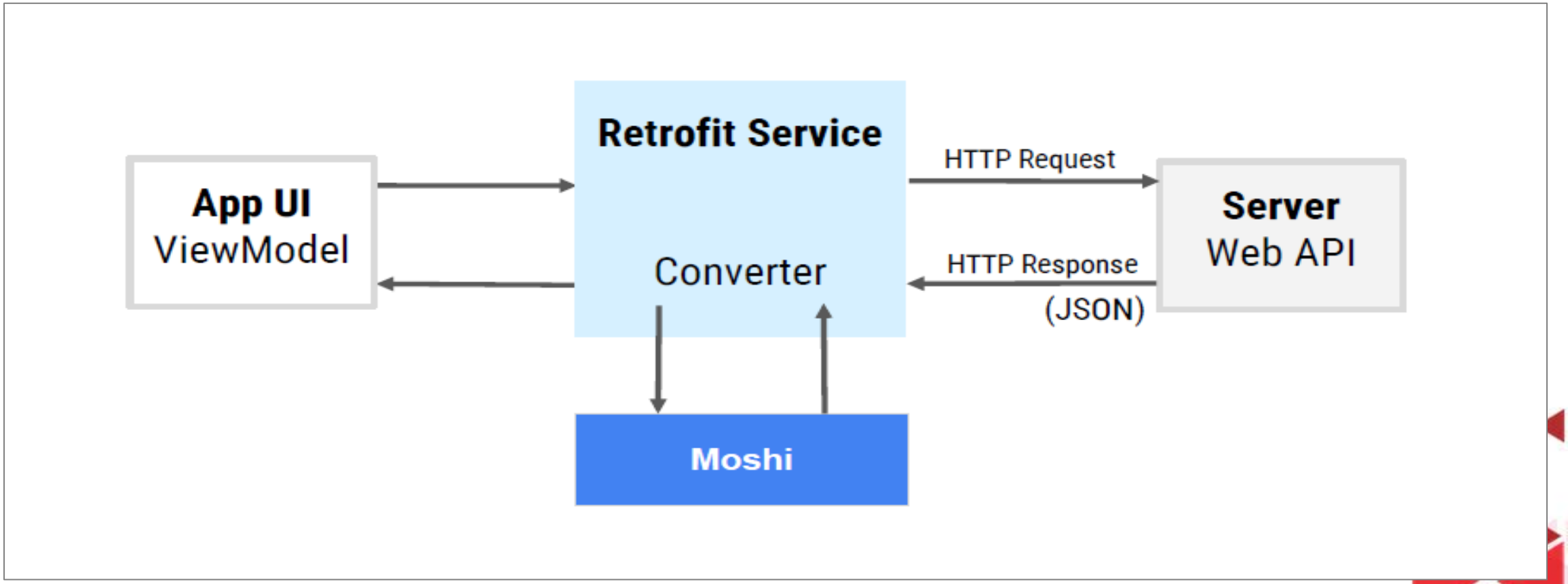
EXTERNAL LIBRARIES

- External libraries, or third-party libraries, are like extensions to the core Android APIs. Using open source, community-developed and maintained libraries can be a huge timesaver.

- However, you must choose these libraries wisely because your app is ultimately responsible for what the code does in these libraries.

- Example of a well-supported and maintained library:
  - ✓ Retrofit: A type-safe HTTP client for Android and Java.
  - ✓ Moshi: A modern JSON library for Kotlin and Java.
  - ✓ Coil: An image loading library for Android backed by Kotlin Coroutines.

RETROFIT X MOSHI



RETROFIT X MOSHI

```
private const val BASE_URL = "https://raw.githubusercontent.com/" +
        "indraazimi/mobpro1-compose/static-api/"

private val moshi = Moshi.Builder()
    .add(KotlinJsonAdapterFactory())
    .build()

private val retrofit = Retrofit.Builder()
    .addConverterFactory(MoshiConverterFactory.create(moshi))
    .baseUrl(BASE_URL)
    .build()

interface HewanApiService {
    @GET("static-api.json")
    suspend fun getHewan(): List<Hewan>
}
```

COIL

```
Box(...) {
    AsyncImage(
        model = ImageRequest.Builder(LocalContext.current)
            .data(HewanApi.getHewanUrl(hewan.imageId))
            .crossfade(true)
            .build(),
        contentDescription = stringResource(R.string.gambar, hewan.nama),
        contentScale = ContentScale.Crop,
        modifier = Modifier.fillMaxWidth()
    )
    Column(...) {
        Text(text = hewan.nama)
        Text(text = hewan.namaLatin)
    }
}
```

REFERENSI

- Permissions on Android
  https://developer.android.com/guide/topics/permissions/overview
- Perform network operations overview
  https://developer.android.com/develop/connectivity/network-ops
- Retrofit: https://square.github.io/retrofit/
- Moshi: https://github.com/square/moshi
- Coil: https://coil-kt.github.io/coil/