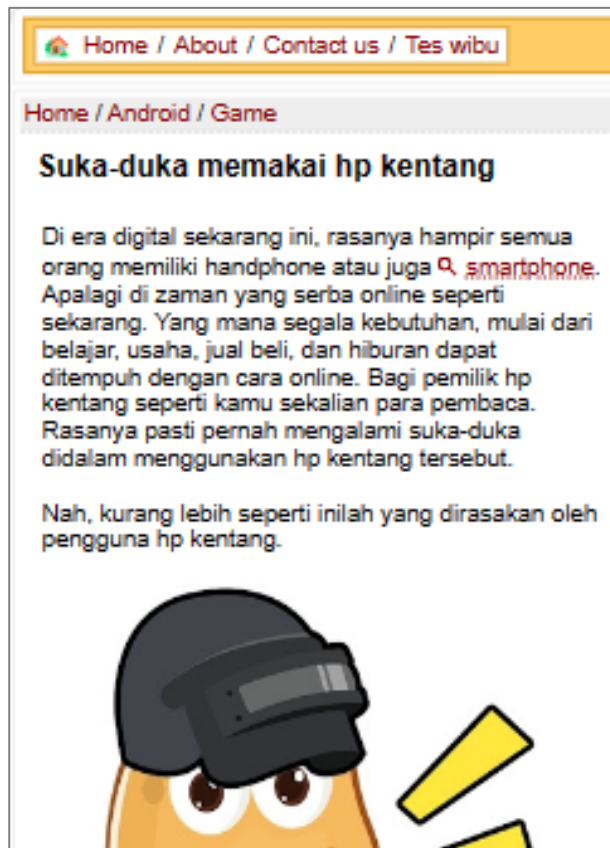


# INTERNET BEST PRACTICES

Tim Dosen Mobpro 1

D3 Rekayasa Perangkat Lunak Aplikasi  
Fakultas Ilmu Terapan

# SOME CHALLENGING ISSUES



# BUILD FOR BILLIONS

The markets with the fastest growing internet and smartphone penetration can have some challenging issues, such as:

- Slow, intermittent, or expensive connectivity.
- Devices with screens, memory, and processors that may be less capable than devices in other markets.
- Limited opportunities to recharge batteries during the day.

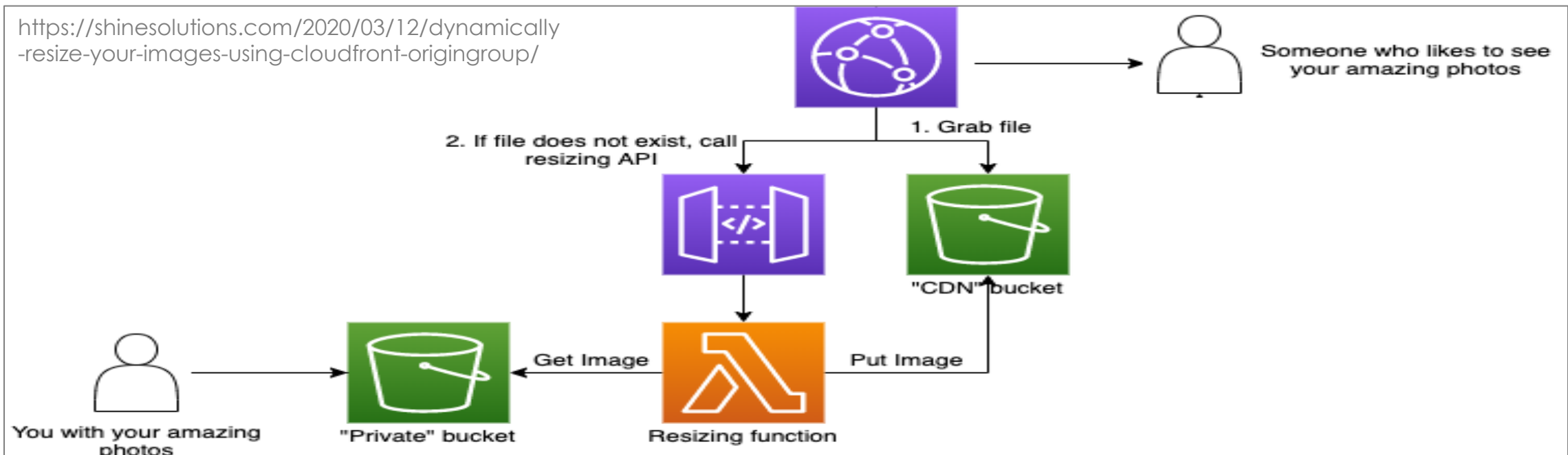
Apps that use memory, power, and network bandwidth efficiently will perform better in any market and produce a **better experience** for all users.

# BUILD FOR BILLIONS

- **Connectivity and data cost:** Provide a better experience for users connected to slower networks. Focus on optimizing images, optimizing networking, and fine-tuning data transfer.
- **Device capability:** Support devices with capabilities that may be different from those you usually develop for. Consider different screen sizes, backward compatibility, and efficient memory use.
- **Battery consumption:** Help preserve battery life. Follow the best practices for power management and benchmarking to ensure sure that your app isn't draining the battery unnecessarily.

# OPTIMIZE IMAGES

Have your apps request images based on the user's device specification, and your server provide **dynamically sized images**.




## Cara resize image di PHP, buka kembali materi **Pemrograman Berbasis Web 1**

← → ↺ <https://github.com/indraazimi/basa-sunda/tree/asset>






indraazimi / basa-sunda

<> Code Issues Pull requests Actions

 **basa-sunda** Public Pin Unwatch 1

asset 2 Branches 0 Tags  Add file <> Code

This branch is 1 commit ahead of, 8 commits behind master. Contribute

Commit	Message	SHA	Time
 <b>indraazimi</b>	Menambahkan aset Basa Sunda	b60a1bd	last year
 BasaSunda-v1.apk	Menambahkan aset Basa Sunda		last year
 BasaSunda-v2.apk	Menambahkan aset Basa Sunda		last year
 getImage.php.png	Menambahkan aset Basa Sunda		last year
 images.zip	Menambahkan aset Basa Sunda		last year



# OPTIMIZE IMAGES

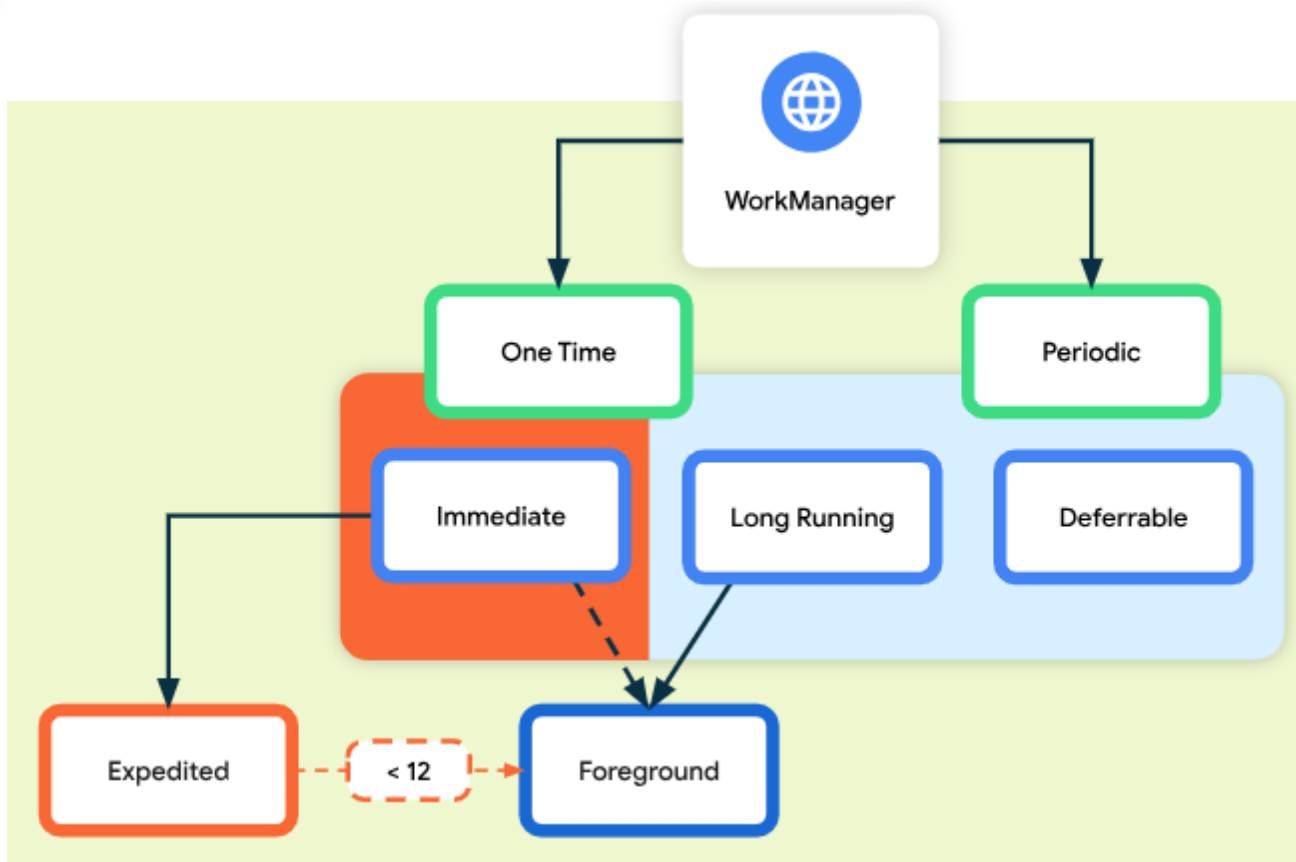
- User experience degrades when users have to wait for images to download. Consider making image size requests based on network type or network quality; this size could be smaller than the target rendering size.
- Serve WebP files over the network to reduce image load times and save network bandwidth. A WebP file is often smaller in size than its PNG and JPG counterparts, with at least the same image quality.
- Your app should not fetch any image more than once. Image loading libraries can fetch the image and cache it. Because images are cached, these libraries return the local copy the next time an image is requested.

# OPTIMIZE NETWORKING

- In rural location and less affluent areas, it's common for devices to lose network connectivity. Creating a useful offline state means users can interact with your app at all times.
- Example of this is an email client that allows users to compose, send, read, move, and delete existing emails even when the device is offline. Or a chat app that have similar features.
- Do this by storing data locally in a database using Room persistence library, caching data, and queuing outbound requests to action when connectivity is restored using WorkManager.



# WORK MANAGER



WorkManager is the primary recommended API for background processing.

Work is persistent: it remains scheduled through app restarts and system reboots.

# WORK MANAGER FEATURES

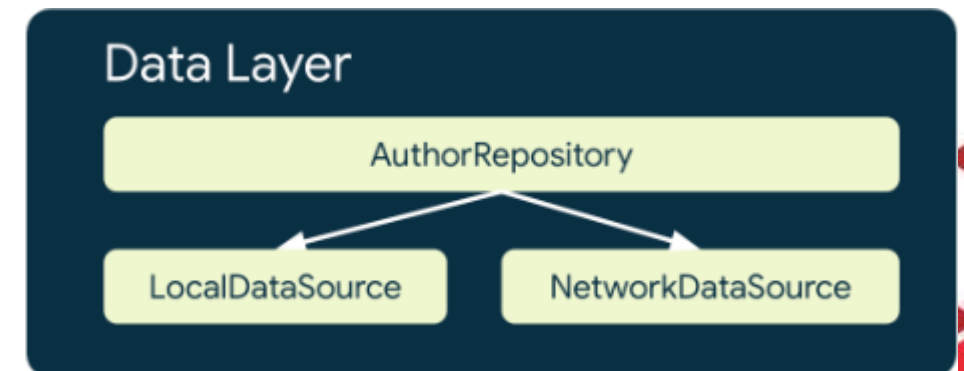
- **Work constraints:** Define the optimal conditions for your work to run. For example, run only when the device is on an unmetered network, when the device is idle, or when it has sufficient battery.
- **Robust scheduling:** Schedule work to run one-time or repeatedly using flexible scheduling windows. WorkManager adheres to power-saving features and best practices like Doze mode.
- **Flexible retry policy:** Sometimes work fails. WorkManager offers flexible retry policies, including a configurable exponential backoff policy with a default delay of 30 seconds.

# RELATIONSHIP TO OTHER APIS

API	Recommended for	Relationship to WorkManager
Coroutines	All asynchronous work that doesn't need to be persistent.	Coroutines are the standard means of leaving the main thread in Kotlin. However, they leave memory once the app closes. For persistent work, use WorkManager.
AlarmManager	Alarms only.	Unlike WorkManager, AlarmManager wakes a device from Doze mode. It is therefore not efficient in terms of power and resource management. Only use it for precise alarms or notifications such as calendar events — not background work.

# OFFLINE-FIRST APP

- **Offline-first app:** an app that is able to perform all, or a critical subset of its core functionality without access to the internet.
- **Lazy writes:** Write to the local data source first, then queue the write to notify the network at the earliest convenience.
- An offline-first app has a minimum of **2 data sources** for every repository that utilizes network resources:
  - The local data source
  - The network data source



# REFERENSI

- Build for Billions  
<https://developer.android.com/docs/quality-guidelines/build-for-billions>
- Schedule tasks with WorkManager  
<https://developer.android.com/topic/libraries/architecture/workmanager>
- Build an Offline-First App  
<https://developer.android.com/topic/architecture/data-layer/offline-first>