

1. Bagian kode yang menangani penyimpanan gambar berdasarkan URI adalah method `generatePhotoUri` di dalam class `ProviderFileManager.kt`. Method ini menentukan lokasi fisik file (menggunakan `getExternalFilesDir`) dan menghasilkan URI menggunakan `FileHelper`.

Potongan kode:

```
fun generatePhotoUri(time: Long): FileInfo {  
    val name = "img_${time}.jpg"  
    // Get the file object  
    // The file will be stored in the folder defined in  
    // file_provider_paths.xml  
    val file = File(  
        context.getExternalFilesDir(fileHelper.getPicturesFolder()),  
        name  
    )  
    return FileInfo(  
        fileHelper.getUriFromFile(file),  
        file,  
        name,  
        fileHelper.getPicturesFolder(),  
        "image/jpeg"  
    )  
}
```

2. Penjelasan atribut pada `FileInfo.kt`:

- Attribute pertama (`uri`): Merujuk pada Content URI sementara yang dihasilkan oleh `FileProvider` (misalnya: `content://com.example.lab_week_11_b.camera/photos/...`). URI ini

digunakan untuk memberikan akses sementara (permission) kepada aplikasi kamera eksternal agar bisa menulis data ke file yang kita buat.

- Attribute keempat (relativePath): Merujuk pada nama folder kategori standar di Android (seperti Environment.DIRECTORY_PICTURES atau "Pictures"). Nilai ini digunakan saat mendaftarkan metadata ke MediaStore agar gambar dikategorikan dengan benar di galeri sistem.

3. [Bonus] Urutan kronologis pengambilan gambar hingga tersimpan di MediaStore:

- a. User menekan tombol "Photo" di MainActivity.
- b. MainActivity mengecek permission storage. Jika diizinkan, method openImageCapture() dipanggil.
- c. openImageCapture() memanggil providerFileManager.generatePhotoUri().
- d. generatePhotoUri() membuat file kosong di folder aplikasi dan menghasilkan URI via FileProvider.
- e. MainActivity meluncurkan aplikasi kamera bawaan (takePictureLauncher.launch) dengan membawa URI tersebut.
- f. User mengambil foto. Aplikasi kamera menyimpan data foto ke URI yang diberikan tadi.
- g. Aplikasi kamera tertutup, callback registerForActivityResult di MainActivity terpanggil.
- h. Di dalam callback, method providerFileManager.insertImageToStore(photoInfo) dijalankan.
- i. insertImageToStore() menyiapkan data metadata (ContentValues) menggunakan MediaContentHelper.
- j. Method insertToStore() dipanggil (di background thread).
- k. ContentResolver membuat entri baru di MediaStore (.insert) dan mendapatkan URI tujuan baru.
- l. Data dari file sementara (URI awal) disalin ke lokasi MediaStore (URI baru) menggunakan IOUtils.copy.