



DIGITAL  
TALENT  
SCHOLARSHIP



# VOCATIONAL SCHOOL GRADUATE ACADEMY

## Mobile Programmer

Pertemuan #8- : Merancang Database dan Data Persistence pada Mobile Data



KOMINFO



#JADIJAGOANDIGITAL

Badan Penelitian dan Pengembangan Sumber Daya Manusia

# PROFIL PENGAJAR



Jabatan Akademik / Kepala LAB Prodi Teknik Komputer

Latarbelakang Pendidikan Pengajar

- S1– STMIK Budi Darma (Skripsi : Aps Mobile Kompresi SMS)
- S2– Universitas Putra Indonesia YPTK Padang (Tesis : Aps Mobile Security SMS)

Riwayat Pekerjaan

- Dosen Tetap Politeknik Negeri Medan
- Trainer Pemrograman Java dan Mobile, Networking, Cyber Security, OS Server
- Konsultan Bidang Aplikasi, Networking, Cyber Security dan Server
- CEO PT. Nusa Tirta Teknologi

Sertifikat Kompetensi :

- Program : Senior Programmer (BNSP)
- Networking : Mikrotik, CISCO
- Server : Windows Server, Redhat
- Project : Comptia Project +

Contact Pengajar

Ponsel :-

Email : [azanuddin@polmed.ac.id](mailto:azanuddin@polmed.ac.id)

# Deskripsi Singkat

## Deskripsi Singkat mengenai Topik

Topik ini akan membahas tentang mendesain internal storage pada aplikasi berbasis mobile dan mendesain eksternal storage pada aplikasi berbasis mobile. Topik ini akan membahas tentang mendesain internal storage pada aplikasi berbasis mobile dan mendesain eksternal storage pada aplikasi berbasis mobile.

## Tujuan Pelatihan

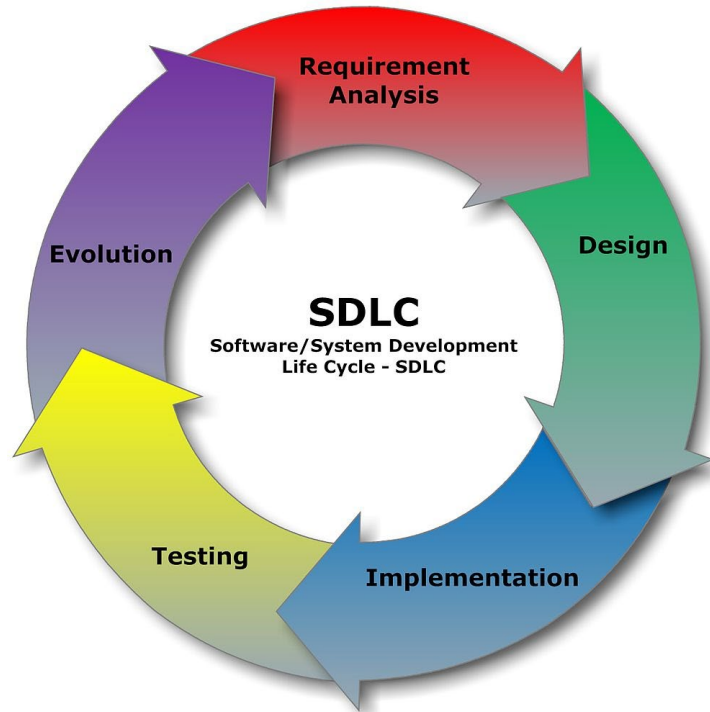
Setelah pertemuan selesai peserta pelatihan mampu:

1. menjelaskan konseptual internal dan external storage pada aplikasi berbasis mobile
2. menjelaskan internal dan external storage berdasarkan spesifikasi device mobile dan operating system berbasis mobile
3. merancang media penyimpanan internal dan external dengan proses create, read, update, delete dan data berjalan dibangun sesuai kebutuhan

## Materi Yang akan disampaikan:

1. Data Storage
2. Internal Storage
3. External Storage
4. File

# Pengenalan SDLC



- **Requirement Analysis:** Penggalan kebutuhan aplikasi termasuk kebutuhan fungsional dan non fungsionalnya.
- **Design:** Perancangan aplikasi termasuk perancangan data, perancangan kelas, desain UI, dlsb.
- **Implementasi:** Dapat juga dikatakan tahap coding / pembuatan source code aplikasi
- **Testing:** Tahapan untuk menguji kesiapan aplikasi
- **Evolution:** Tahap pengembangan aplikasi yang berkelanjutan termasuk penambahan fitur, bug fixing, penyesuaian kebutuhan, dlsb.

# Data Storage

- ❖ Penyimpanan data pada android terdiri dari dua jenis data, yaitu :
  1. data *temporary*
  2. data *persistence*.
- ❖ Data *temporary* umumnya diproses melalui *internal storage* dalam bentuk *cache*.
- ❖ Data *persistence* disimpan secara *permanen* dalam suatu file.

# Mengapa *Data Persistence* diperlukan?

Android dapat di-*shutdown* dan di-*restart* kapanpun dibutuhkan, namun bagaimana ketika:

- merotasi *screen*
- mengubah Bahasa
- aplikasi berjalan secara *background process* dan disimpan dalam *short-in memory*
- menekan tombol Back

Apakah yang akan terjadi? data hilang!

# Lalu Bagaimana Solusinya?

- ❖ Android menyediakan beberapa solusi untuk menangani data *persistence*, tergantung dari sifatnya apakah *private* atau *public*.
- ❖ Beberapa solusi yang dipilih harus sesuai dengan kebutuhan spesifiknya, seperti apakah data yang disimpan akan bersifat *private* atau boleh diakses secara bebas (*public*) serta berapa banyak *space* yang dibutuhkannya.
- ❖ Android juga menyediakan cara untuk membuka data privat ke aplikasi lain (apabila diperlukan), melalui **Content Provider**.

# Beberapa Opsi *Data Storage*

- ❖ *Shared Preferences*: menyimpan data primitive berupa *key-value pairs*.
- ❖ *Internal Storage*: menyimpan data *private* pada *device memory*.
- ❖ *External Storage*: menyimpan data public pada *shared external storage*.
- ❖ *Database*: menyimpan data struktur pada sebuah database
- ❖ *Network Connection*: menyimpan data pada web server
- ❖ *Bundle Class*: memetakan data string dalam bentuk map
- ❖ *File IO*: menyimpan data berupa file pada internal storage



# Android File System

- **Internal Storage** – direktori *private* yang hanya diakses pada aplikasi itu sendiri.
- **External Storage** – direktori *public* dapat diakses secara lebih luas pada aplikasi itu sendiri dan aplikasi lainnya.

Aplikasi dapat mengakses struktur direktori, Struktur dan operasinya mirip seperti yang digunakan pada **Linux** dan **java.io**

# Karakteristik dari Internal Storage

- ❖ Selalu tersedia (dapat diakses).
- ❖ Menggunakan *file system* dari *device*.
- ❖ Hanya bisa diakses pada aplikasi itu sendiri, kecuali jika secara eksplisit diatur agar dapat dibaca / ditulis.
- ❖ Ketika aplikasi di-uninstall, sistem akan menghapus semua file aplikasi dari *internal storage*.

# Karakteristik dari Eksternal Storage

- ❖ Tidak selalu tersedia (dapat dihapus)
- ❖ Menggunakan *file system* dari *device* atau dari penyimpanan fisik eksternal seperti SD Card.
- ❖ *World-readable*, semua aplikasi dapat membaca file tersebut.
- ❖ Ketika aplikasi di-uninstall, *system* tidak akan menghapus file *private* dari aplikasi tersebut.

# Kapan Kita Perlu Internal dan Eksternal Storage?

**Internal storage** paling baik digunakan jika:

- Tidak ada yang boleh mengakses file selain aplikasi itu sendiri.

**Eksternal storage** paling baik digunakan jika:

- File tidak memerlukan batasan akses (semua *public*).
- File dapat di-share ke aplikasi lain.
- Membolehkan pengguna untuk mengakses file melalui komputer (via *USB Connection*).

# Internal Storage

- File bersifat private
- Aplikasi akan selalu meminta izin untuk read/write data
- Direktori penyimpanan *persistence* – `getFilesDir()`
- Direktori penyimpanan *temporary* – `getCacheDir()`

# Contoh Pembuatan File

```
File file = new File(context.getFilesDir(), filename);
```

Untuk fungsi-fungsi pengelolaan filenya, menggunakan standard [java.io](#) *file operator* atau *stream*.

# Langkah menulis file private ke Internal Storage

Untuk membuat dan menuliskan file *private* ke *internal storage* :

1. Panggil `openFileOutput()` dengan nama file dan mode pengoperasiannya. Proses ini mengembalikan `FileOutputStream`.
2. Tulis pada file melalui method `write()`.
3. Tutup aliran (***stream***) melalui method `close()`.

# Contoh Kode Program untuk Menulis ke File

```
String filename = "myfile";  
String fileContents = "Hello world!";  
FileOutputStream outputStream;  
  
try {  
    outputStream = openFileOutput(filename,  
Context.MODE_PRIVATE);  
    outputStream.write(fileContents.getBytes());  
    outputStream.close();  
} catch (Exception e) {  
    e.printStackTrace();  
}
```



- `MODE_PRIVATE` akan membuat file (atau mengganti/*replace* file dengan nama yang sama) dan membuatnya *private* untuk aplikasi anda. Mode lain yang tersedia antara lain :
  - `MODE_APPEND`
  - `MODE_WORLD_READABLE` \*
  - `MODE_WORLD_WRITEABLE` \*

Catatan : \* sudah tidak digunakan lagi sejak API level 17 (dimulai dari android N) karena penggunaannya akan berdampak pada *SecurityException*.

# Langkah membaca file *private* dari *internal storage*

Untuk membaca file *private* dari *internal storage* :

1. Panggil `openFileInput()` dan memberikan nama file yang ingin dibaca.  
Proses ini mengembalikan `FileInputStream`
2. Baca byte dari file dengan method `read()`
3. Tutup aliran (*stream*) dengan method `close()`

# Contoh Kode Program untuk Membaca dari File

```
String FILENAME = "hello_file.txt";  
FileInputStream fin = openFileInput(FILENAME);  
StringBuffer fileContent = new StringBuffer("");  
  
byte[] buffer = new byte[1024];  
while ((n=fin.read(buffer)) != -1) {  
    fileContent.append(new String(buffer, 0, n));  
}  
fin.close();
```

# Menyimpan File Cache

- `getCacheDir()` – digunakan untuk membuka file pada direktori *internal storage*.
- `getCacheExternalDir()` – digunakan untuk membuka *cache* pada *external storage*.

# Contoh kode untuk Menuliskan File *Cache*

- `createTempFile()` akan membuat file pada direktori *cache* yang tersimpan secara privat pada aplikasi.

```
private File getTempFile(Context context, String url) {  
    File file;  
    try {  
        String fileName = Uri.parse(url).getLastPathSegment();  
        file = File.createTempFile(fileName, null, context.getCacheDir());  
    } catch (IOException e) {  
        // Error while creating file  
    }  
    return file;  
}
```

# Beberapa Method Penting Lainnya

- `getFilesDir()` – mendapatkan path absolut ke direktori file pada *internal storage*.
- `getDir()` – membuat (atau membuka yang sudah ada) direktori pada *internal storage*.
- `deleteFile()` – menghapus file yang disimpan pada *internal storage*.
- `fileList()` – mengembalikan daftar file yang saat ini telah disimpan dalam *internal storage*.

# Eksternal Storage

Android menyediakan metode penyimpanan file external pada ponsel atau tablet yang terbagi berdasarkan beberapa jenis, yaitu:

1. Untuk ponsel Android generasi awal biasanya hanya dibagi menjadi dua bagian, yaitu:
  - Internal storage
  - SD card

# Eksternal Storage

2. Untuk Smartphone yang tidak memiliki slot SD card juga dibagi menjadi dua bagian, yaitu:
  - Internal/System storage,
  - Phone/USB Storage.
3. Untuk smartphone dan tablet Android saat ini pada umumnya dibagi menjadi tiga bagian, yaitu:
  - Internal/System storage,
  - Phone/USB Storage,
  - SD card.



# Perbandingan Storage Antar Platform

## ❖ Perbandingan Storage pada Android dan iPhone

- **iPhone**

iPhone memiliki kapasitas internal lebih tinggi dari android dan partisi untuk OS lebih fleksibel namun tidak support untuk external storage atau tambahan memory, kapasitas nya adalah 8GB,16GB, 32GB, 64GB, 128GB, 256GB belum termasuk partisi untuk OS yang memakan sekitar 1,5GB – 2GB

# Perbandingan Storage Antar Platform

- **Android**

Android memiliki kapasitas internal yang lebih kecil dari iOS dengan partisi untuk OS biasanya memakan separuh dari kapasitas internalnya, memiliki opsi external storage, kapasitas internal nya mulai dari 4GB, 8GB, 16GB, 32GB, 64GB belum termasuk partisi OS yang memakan separuh dari internal storage.

# Konfigurasi pada AndroidManifest.xml

Agar bisa membaca atau menulis file pada penyimpanan eksternal, aplikasi harus memperoleh `READ_EXTERNAL_STORAGE` atau izin sistem `WRITE_EXTERNAL_STORAGE` yang di daftarkan pada `AndroidManifest.xml`. Misalnya seperti baris kode dibawah:

```
<manifest>

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

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

</manifest>
```

# Fungsi Penting dalam Penyimpanan Eksternal

- `getExternalStorageState()`: untuk memeriksa apakah media tersebut tersedia.
- `getExternalStorageDirectory()`: nilai dari fungsi ini menghasilkan lokasi direktori penyimpanan eksternal / eksternal utama.
- `getExternalStoragePublicDirectory(String type)`: nilai dari fungsi ini menghasilkan lokasi direktori penyimpanan eksternal / eksternal utama sesuai dari isi Argumen `type` yang ada pada fungsi tersebut

# External Storage State

Dibawah adalah beberapa contoh baris kode penggunaan *External Storage State*

```
/* Periksa apakah penyimpanan eksternal
tersedia untuk membaca dan menulis */

public boolean isExternalStorageWritable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state)) {
        return true;
    }
    return false;
}
```

```
/* Periksa apakah penyimpanan eksternal
setidaknya tersedia untuk dibaca */

public boolean isExternalStorageReadable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state) ||
        Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
        return true;
    }
    return false;
}
```

# Jenis - Jenis Status External Storage

Adapun nilai balik status dari fungsi `getExternalStorageState()` di antaranya :

- `MEDIA_UNKNOWN` : kartu SD Card tidak dapat dikenali
- `MEDIA_REMOVED` : status tidak ada kartu SD Card yang tersedia
- `MEDIA_UNMOUNTED` : Status penyimpanan jika media ada tetapi tidak dipasang.
- `MEDIA_MOUNTED` : Status penyimpanan jika media ada dan terpasang dengan akses baca / tulis.
- `MEDIA_MOUNTED_READ_ONLY` : Status penyimpanan jika media ada dan terpasang dengan akses hanya baca.

# External Files Dir

```
public void buatFile() {  
    String cashback = "tuliskan isi file";  
    String state = Environment.getExternalStorageState();  
    //external storage availability check  
    if (!Environment.MEDIA_MOUNTED.equals(state)) {  
        return;  
    }  
    //Pembuatan file yang akan disimpan dengan direktori Eksternal  
    File file = new File(Environment.getExternalStorageDirectory(), child: "DemoFile.txt");  
    FileOutputStream outputStream = null;  
    try {  
        file.createNewFile();  
        //Argumen kedua konstruktor FileOutputStream menunjukkan apakah  
        // akan menambahkan isi data atau membuat file baru jika sudah ada  
        outputStream = new FileOutputStream(file, append: true);  
        outputStream.write(cashback.getBytes());  
        outputStream.flush();  
        outputStream.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

# External Storage Public Directory

```
public void buatFile() {  
    String cashback = "tuliskan isi file";  
    String state = Environment.getExternalStorageState();  
    //external storage availability check  
    if (!Environment.MEDIA_MOUNTED.equals(state)) {  
        return;  
    }  
    //Pembuatan file yang akan disimpan di direktori Eksternal  
    File file = new File(Environment.getExternalStoragePublicDirectory(  
        Environment.DIRECTORY_DOCUMENTS), child: "namafile.txt");  
    FileOutputStream outputStream = null;  
    try {  
        file.createNewFile();  
        //Argumen kedua konstruktor FileOutputStream menunjukkan apakah  
        // akan menambahkan isi data atau membuat file baru jika sudah ada  
  
        outputStream = new FileOutputStream(file, append: true);  
        outputStream.write(cashback.getBytes());  
        outputStream.flush();  
        outputStream.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

Penggunaan public directory  
Hanya berlaku untuk versi OS  
(19) Kitkat keatas



# Jenis - Jenis Direktori Publik

Adapun nilai-nilai static yang sering digunakan dari fungsi `getExternalStoragePublicDirectory(String type)` di antaranya:

- `DIRECTORY_DOCUMENTS` : Direktori standar tempat meletakkan dokumen yang telah dibuat oleh pengguna.
- `DIRECTORY_DOWNLOADS` : Direktori standar tempat meletakkan file yang telah diunduh oleh pengguna.
- `DIRECTORY_MOVIES` : Direktori standar untuk menempatkan film yang tersedia bagi pengguna.
- `DIRECTORY_PICTURES` : Direktori standar tempat meletakkan gambar yang tersedia bagi pengguna.

# Membuat Fungsi Create,Read Update dan Delete

Pada penyimpanan eksternal dapat melakukan beberapa hal diantaranya :

- Create
- Read
- Update
- dan Delete

# Create File

```
public void buatFile() {  
    String cashback = "tuliskan isi file";  
    String state = Environment.getExternalStorageState();  
    //external storage availability check  
    if (!Environment.MEDIA_MOUNTED.equals(state)) {  
        return;  
    }  
    //Pembuatan file yang akan disimpan dengan direktori Eksternal  
    File file = new File(Environment.getExternalStorageDirectory(), child: "DemoFile.txt");  
    FileOutputStream outputStream = null;  
    try {  
        file.createNewFile();  
        //Argumen kedua konstruktor FileOutputStream menunjukkan apakah  
        // akan menambahkan isi data atau membuat file baru jika sudah ada  
        outputStream = new FileOutputStream(file, append: true);  
        outputStream.write(cashback.getBytes());  
        outputStream.flush();  
        outputStream.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

# Read File

```
public void bacaFile() {  
    File sdcard = Environment.getExternalStorageDirectory();  
    File file = new File(sdcard, FILENAME);  
  
    if(file.exists()) {  
        StringBuilder text = new StringBuilder();  
  
        try {  
            BufferedReader br = new BufferedReader(new FileReader(file));  
  
            String line = br.readLine();  
  
            while (line != null) {  
                text.append(line);  
                line = br.readLine();  
            }  
            br.close();  
        } catch (IOException e) {  
            System.out.println("Error " + e.getMessage());  
        }  
        textBaca.setText(text.toString());  
    }  
}
```

File diubah menjadi stream menggunakan FileReader sehingga dapat dibaca dalam BufferedReader, setelah BufferedReader mempunyai value dari FileReader, maka data tersebut dibaca baris perbaris menggunakan StringBuilder.

# Delete File

```
void hapusFile() {  
    File file = new File(Environment.getExternalStorageDirectory(), FILENAME);  
    if (file.exists()) {  
        file.delete();  
    }  
}
```

`file.delete()` mempunyai nilai balik berupa Boolean yang mana jika nilai true, maka data berhasil dihapus

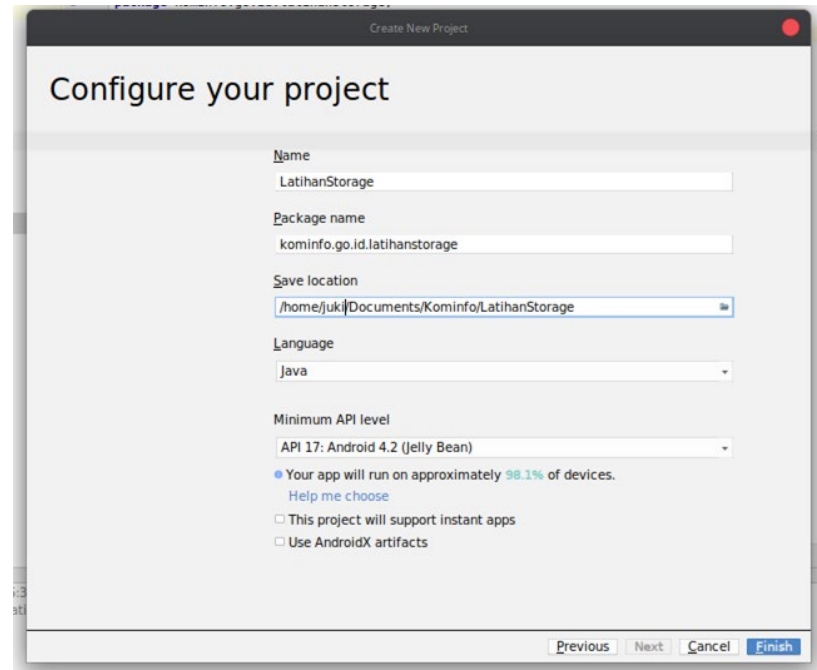
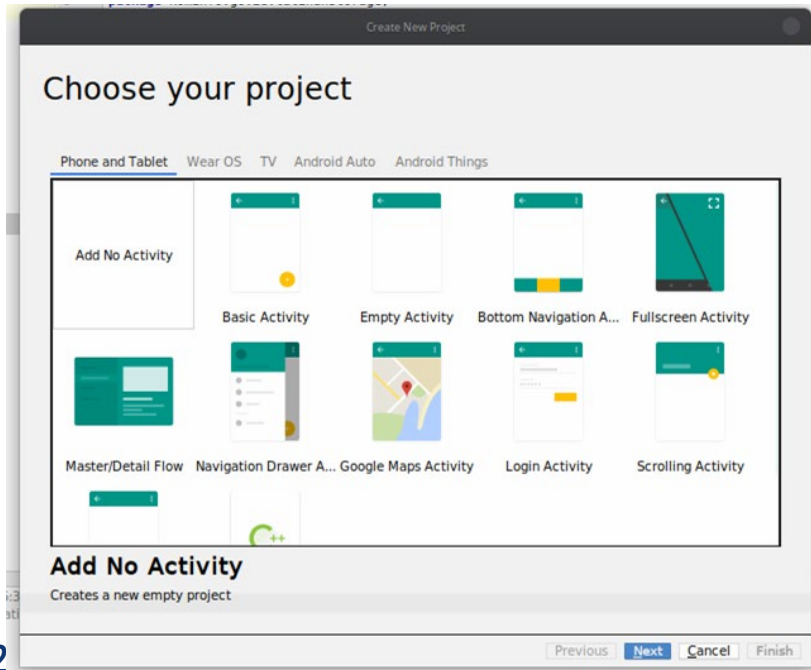
# Latihan 1

Tahapan-tahapan pembuatan Aplikasi pemanfaatan *Internal Storage* :

- **[Langkah 1]** Buatlah sebuah Project Baru Android dan beri nama Aplikasi `LatihanStorage`
- **[Langkah 2]** Buat Sebuah Java Kelas yang mewariskan Kelas Activity, beri nama Kelas `InternalStorageActivity.java`
- **[Langkah 3]** Buat Sebuah *Layout* xml dengan nama `layout_internal.xml`
- **[Langkah 4]** Pada Java Kelas `InternalStorageActivity.java` Tambahkan Baris Fungsi *Create, Read, Update* dan *Delete*.
- **[Langkah 5]** Jalankan Aplikasi.

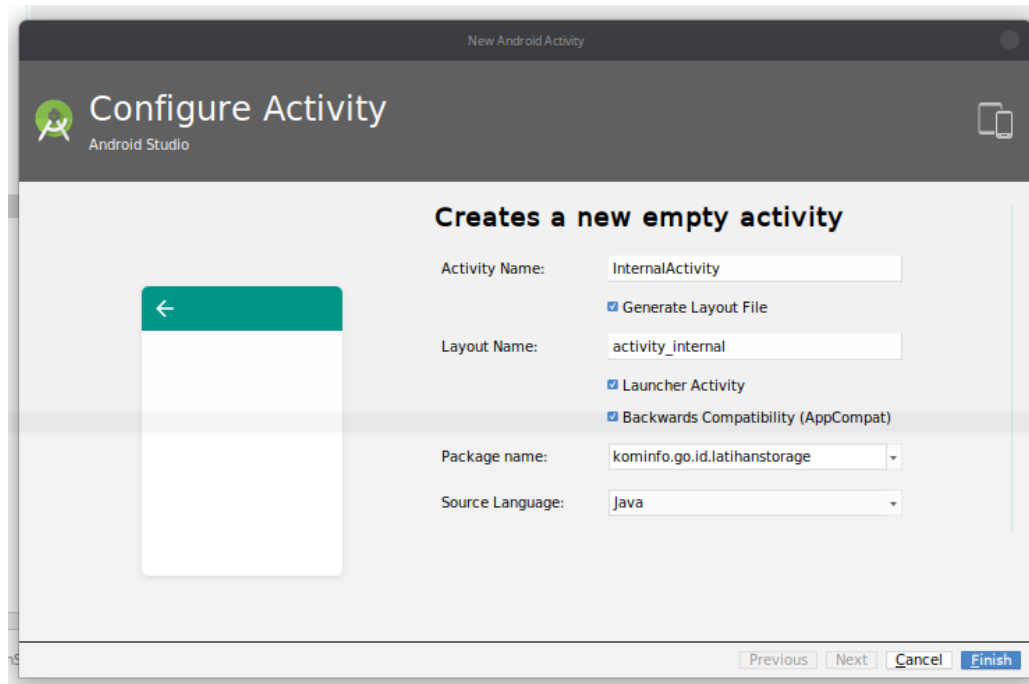
# Latihan 1 – Langkah 1

**[Langkah 1]** Buatlah sebuah Project Baru Android dan beri nama Aplikasi LatihanStorage



# Latihan 1 – Langkah 2

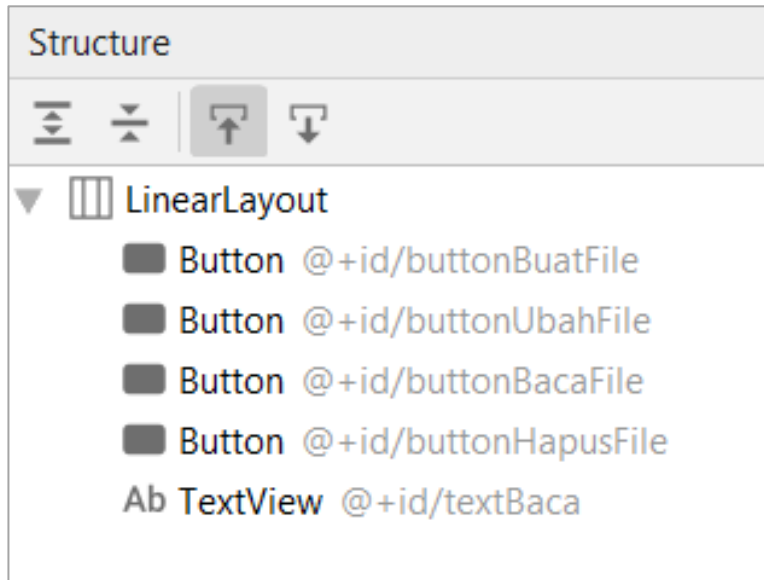
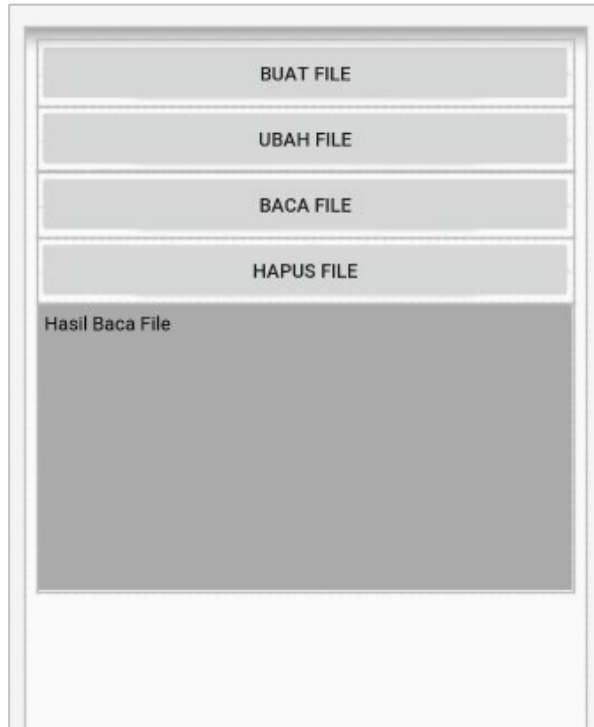
**[Langkah 2]** Buat Sebuah Activity, beri nama Kelas `InternalActivity.java` dan sebuah Layout dengan nama `activity_internal.xml`





# Latihan 1 – Langkah 3

**[Langkah 3]** Membuat sebuah view didalam layout `activity_internal.xml`



# Latihan 1 – Langkah 3

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp"
    android:orientation="vertical">

    <Button
        android:id="@+id/buttonBuatFile"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Buat File" />

    <Button
        android:id="@+id/buttonUbahFile"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Ubah File" />

    <Button
        android:id="@+id/buttonBacaFile"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Baca File" />
```


```
    <Button
        android:id="@+id/buttonHapusFile"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Hapus File" />

    <TextView
        android:id="@+id/textBaca"
        android:layout_width="match_parent"
        android:layout_height="210dp"
        android:background="@android:color/darker_gray"
        android:ems="10"
        android:gravity="top|left"
        android:padding="5dp"
        android:textColorHint="#000"
        android:hint="Hasil Baca File"
        android:inputType="textMultiLine" />

</LinearLayout>
```

# Latihan 1 – Langkah 4

**[Langkah 4]** Pada Java Kelas `InternalActivity.java` Tambahkan baris fungsi *Create, Read, Update* dan *Delete*.



```
package kominfo.go.id.latihanstorage;

import android.os.Environment;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;

public class InternalActivity extends AppCompatActivity implements View.OnClickListener {
    public static final String FILENAME = "namafile.txt";
    Button buatFile, ubahFile, bacaFile, deleteFile;
    TextView textBaca;
```

# Latihan 1 – Langkah 4

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_internal);
    buatFile = findViewById(R.id.buttonBuatFile);
    ubahFile = findViewById(R.id.buttonUbahFile);
    bacaFile = findViewById(R.id.buttonBacaFile);
    deleteFile = findViewById(R.id.buttonHapusFile);
    textBaca = findViewById(R.id.textBaca);

    buatFile.setOnClickListener(this);
    ubahFile.setOnClickListener(this);
    bacaFile.setOnClickListener(this);
    deleteFile.setOnClickListener(this);
}
```

2

```
void buatFile() {
    String isiFile = "Coba Isi Data File Text";
    File file = new File(getFilesDir(), FILENAME);

    FileOutputStream outputStream = null;
    try {
        file.createNewFile();
        outputStream = new FileOutputStream(file, append: true);
        outputStream.write(isiFile.getBytes());
        outputStream.flush();
        outputStream.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

3

# Latihan 1 – Langkah 4

```
void hapusFile() {  
    File file = new File(getFilesDir(), FILENAME);  
    if (file.exists()) {  
        file.delete();  
    }  
}
```

4

```
@Override  
public void onClick(View v) { jalankanPerintah(v.getId()); }  
  
public void jalankanPerintah(int id) {  
    switch (id) {  
        case R.id.buttonBuatFile:  
            buatFile();  
            break;  
        case R.id.buttonBacaFile:  
            bacaFile();  
            break;  
        case R.id.buttonUbahFile:  
            ubahFile();  
            break;  
        case R.id.buttonHapusFile:  
            hapusFile();  
            break;  
    }  
}
```

5

# Latihan 1 – Langkah 4

```
void hapusFile() {  
    File file = new File(getFilesDir(), FILENAME);  
    if (file.exists()) {  
        file.delete();  
    }  
}
```

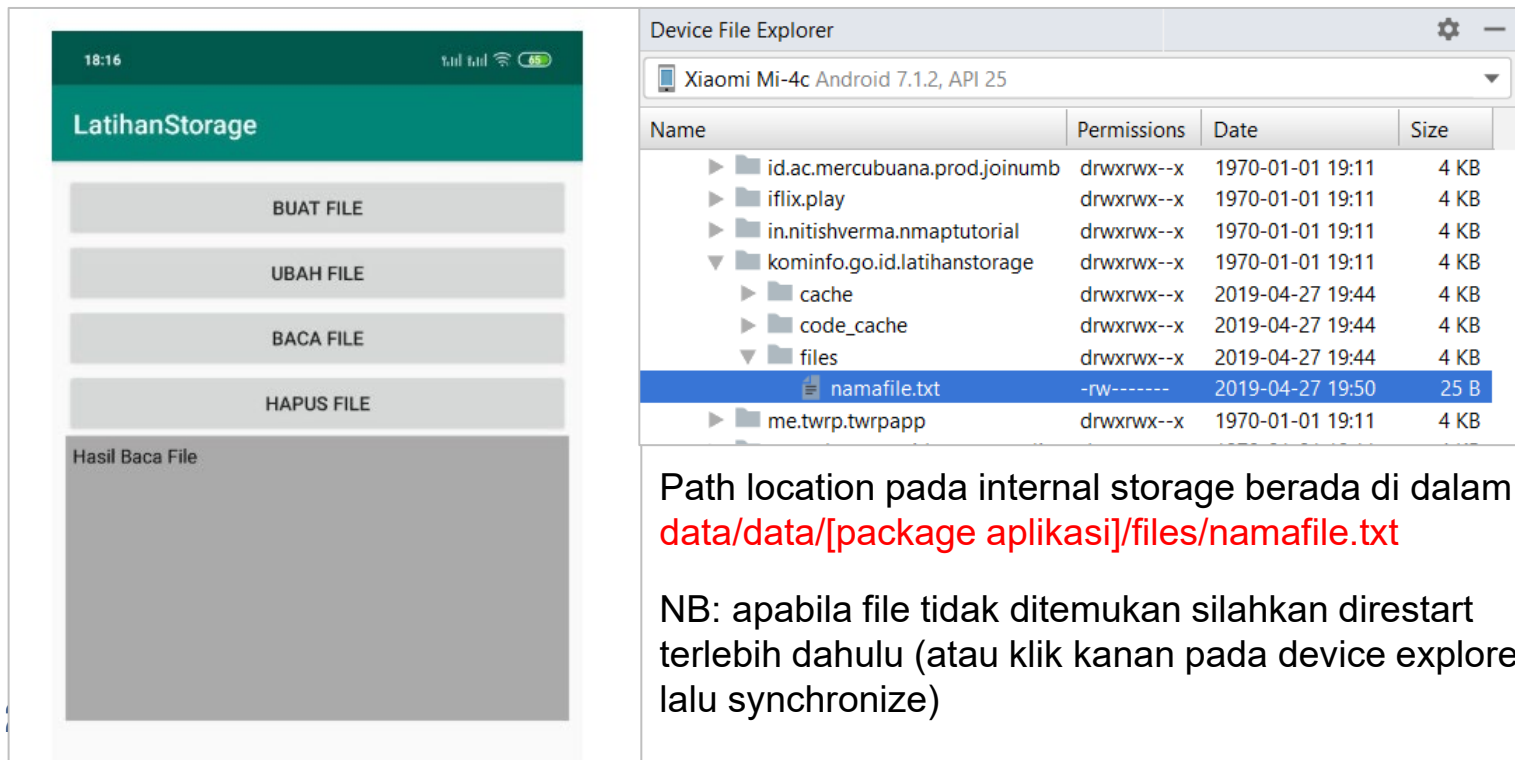
6

```
@Override  
public void onClick(View v) { jalankanPerintah(v.getId()); }  
  
public void jalankanPerintah(int id) {  
    switch (id) {  
        case R.id.buttonBuatFile:  
            buatFile();  
            break;  
        case R.id.buttonBacaFile:  
            bacaFile();  
            break;  
        case R.id.buttonUbahFile:  
            ubahFile();  
            break;  
        case R.id.buttonHapusFile:  
            hapusFile();  
            break;  
    }  
}
```

7

# Latihan 1 – Langkah 5

## [Langkah 5] Jalankan Aplikasi pada Device / Emulator



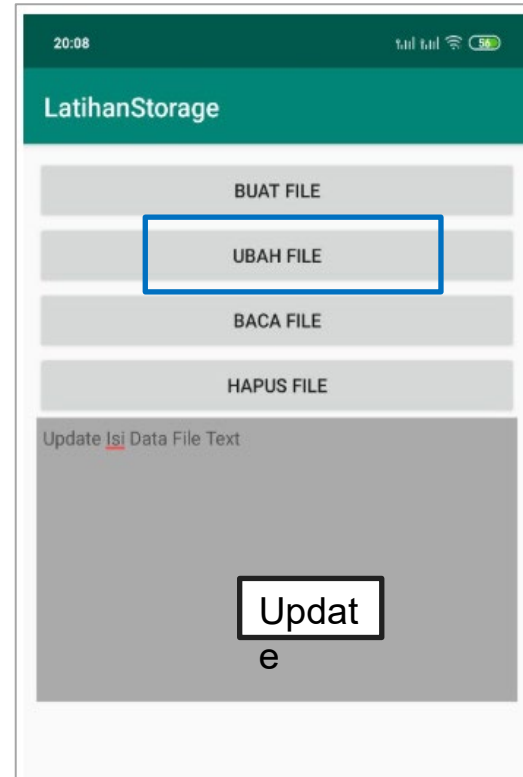
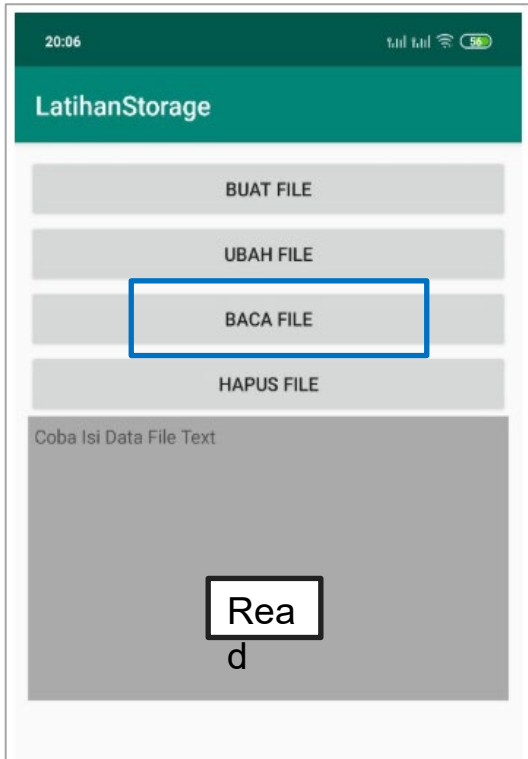
The image shows two side-by-side screenshots. The left screenshot is of an Android application titled "LatihanStorage". It has a green header bar with the title. Below the header, there are four buttons: "BUAT FILE", "UBAH FILE", "BACA FILE", and "HAPUS FILE". At the bottom, there is a grey box labeled "Hasil Baca File". The right screenshot is of the "Device File Explorer" application, showing the internal storage of a "Xiaomi Mi-4c Android 7.1.2, API 25". The file list shows a directory structure where the file "namafile.txt" is located at the path "data/data/[package aplikasi]/files/namafile.txt". The file is highlighted in blue.

Name	Permissions	Date	Size
▶ id.ac.mercubuana.prod.joinumb	drwxrwx--x	1970-01-01 19:11	4 KB
▶ iflix.play	drwxrwx--x	1970-01-01 19:11	4 KB
▶ in.nitishverma.nmaptutorial	drwxrwx--x	1970-01-01 19:11	4 KB
▼ kominfo.go.id.latihanstorage	drwxrwx--x	1970-01-01 19:11	4 KB
▶ cache	drwxrwx--x	2019-04-27 19:44	4 KB
▶ code_cache	drwxrwx--x	2019-04-27 19:44	4 KB
▼ files	drwxrwx--x	2019-04-27 19:44	4 KB
namafile.txt	-rw-----	2019-04-27 19:50	25 B
▶ me.twrp.twrpapp	drwxrwx--x	1970-01-01 19:11	4 KB

Path location pada internal storage berada di dalam **data/data/[package aplikasi]/files/namafile.txt**

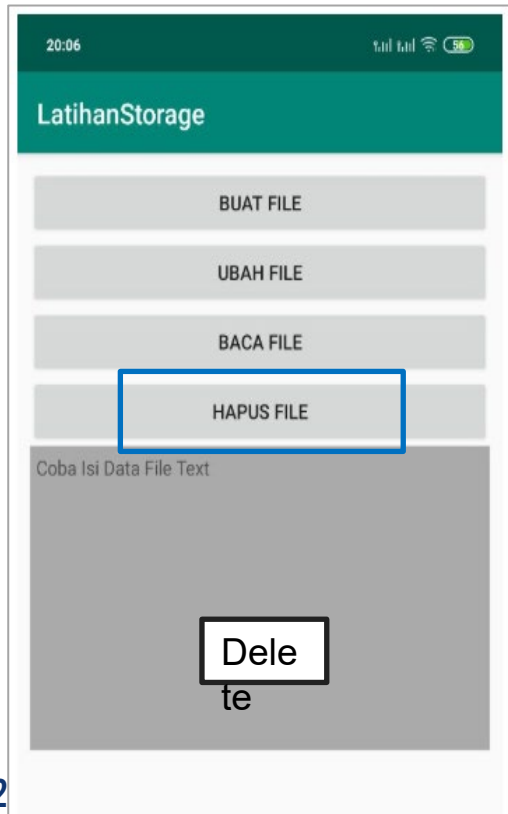
NB: apabila file tidak ditemukan silahkan direstart terlebih dahulu (atau klik kanan pada device explorer lalu synchronize)

# Latihan 1 – Langkah 5





# Latihan 1 – Langkah 5

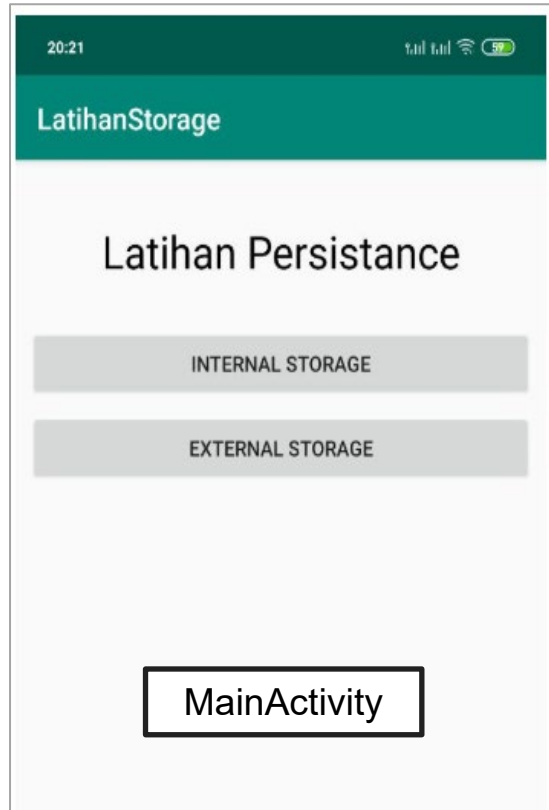


The screenshot shows the Device File Explorer app interface. At the top, there's a header with the text 'Device File Explorer'. Below it, there's a dropdown menu showing 'Xiaomi Redmi Note 7 Android 9, API 28'. Below the dropdown, there's a table with columns: Name, Permissions, Date, and Size. The table lists several files and folders, including 'com.xiaomi.xmsf', 'id.co.babe', 'kominfo.go.id.latihanstorage', 'cache', 'code\_cache', 'files' (highlighted with a blue row), 'org.codeaurora.ims', and 'org.telegram.messenger'.

Name	Permissions	Date	Size
▶ com.xiaomi.xmsf	drwxrwx--x	1970-01-01 22:02	4 KB
▶ id.co.babe	drwxrwx--x	1970-01-01 22:02	4 KB
▼ kominfo.go.id.latihanstorage	drwxrwx--x	1970-01-01 22:02	4 KB
▶ cache	drwxrws--x	2019-04-27 20:06	4 KB
▶ code_cache	drwxrws--x	2019-04-27 20:06	4 KB
▶ files	drwxrwx--x	2019-04-27 20:12	4 KB
▶ org.codeaurora.ims	drwxrwx--x	1970-01-01 22:02	4 KB
▶ org.telegram.messenger	drwxrwx--x	1970-01-01 22:02	4 KB

**File “namafile.txt” pada internal storage telah terhapus**

# Latihan 2



- ❖ Berdasarkan latihan 1 yang telah dilakukan sebelumnya, penyimpanan masih dilakukan secara *internal storage*, selanjutnya silahkan tambahkan fitur agar aplikasi dapat melakukan penyimpanan juga melalui *external storage*.
- ❖ **[Langkah 1]** Tambahkan sebuah *activity* yang memuat 2 *button* yang berfungsi untuk menyimpan data secara *internal* maupun *external* sebagaimana tampilan berikut dengan nama `MainActivity.java`

## Latihan 2

- **[Langkah 2]** Sesuaikan dengan fungsi button yang telah dibuat pada Langkah 1, yakni apabila dipilih tombol internal storage maka akan memunculkan `InternalActivity.java`, sedangkan apabila dipilih tombol external storage akan memunculkan halaman baru yang memiliki tampilan dan fungsi yang sama dengan internal storage (yang membedakan hanya lokasi penyimpanan menggunakan *external storage*).

# Proyek Aplikasi Sederhana Penyimpan Data *Persistence*

- **[Proyek 1]** : Aplikasi Catatan Harian dengan menggunakan file pada Android
- **[Proyek 2]** : Aplikasi Validasi Login yang melakukan pengecekan terhadap data user yang tersimpan pada file Teks.

# Referensi

1. <http://bit.ly/2WaH2bl>
1. <https://developer.android.com/reference/android/os/Environment.html>
2. <https://developer.android.com/reference/java/io/File.html>
3. <https://developer.android.com/guide/topics/data/data-storage?hl=id>
4. <https://developer.android.com/training/data-storage/files#InternalVsExternalStorage>

# Tim Penyusun

- Alif Akbar Fitrawan, S.Pd, M. Kom (Politeknik Negeri Banyuwangi);
- Anwar, S.Si, MCs. (Politeknik Negeri Lhokseumawe);
- Eddo Fajar Nugroho (BPPTIK Cikarang);
- Eddy Tungadi, S.T., M.T. (Politeknik Negeri Ujung Pandang);
- Fitri Wibowo (Politeknik Negeri Pontianak);
- Ghifari Munawar (Politeknik Negeri Bandung);
- Hetty Meileni, S.Kom., M.T. (Politeknik Negeri Sriwijaya) ;
- I Wayan Candra Winetra, S.Kom., M.Kom (Politeknik Negeri Bali) ;
- Irkham Huda (Vokasi UGM) ;
- Josseano Amakora Koli Parera, S.Kom., M.T. (Politeknik Negeri Ambon) ;
- I Komang Sugiarta, S.Kom., MMSI (Universitas Gunadarma) ;
- Lucia Sri Istiyowati, M.Kom (Institut Perbanas) ;
- Maksy Sendiang, ST, MIT (Politeknik Negeri Manado) ;
- Medi Noviana (Universitas Gunadarma) ;
- Muhammad Nashrullah (Politeknik Negeri Batam) ;
- Nat. I Made Wiryana, S.Si., S.Kom., M.Sc. (Universitas Gunadarma) ;
- Rika Idmayanti, ST, M.Kom (Politeknik Negeri Padang) ;
- Rizky Yuniar Hakkun (Politeknik Elektronik Negeri Surabaya) ;
- Robinson A.Wadu, ST., MT (Politeknik Negeri Kupang) ;
- Roslina. M.IT (Politeknik Negeri Medan) ;
- Sukamto, SKom., MT. (Politeknik Negeri Semarang) ;
- Syamsi Dwi Cahya, M.Kom. (Politeknik Negeri Jakarta) ;
- Syamsul Arifin, S.Kom, M.Cs (Politeknik Negeri Jember) ;
- Usmanudin (Universitas Gunadarma) ;
- Wandy Alifha Saputra (Politeknik Negeri Banjarmasin) ;

# #JADIJAGOANDIGITAL TERIMA KASIH



digitalent.kominfo



DTS\_kominfo



digitalent.kominfo



digital talent scholarship