Modul Praktikum MP — #8

Topik	:	**Menggunakan Multimedia**
Deskripsi	:	Praktikum ini membahas pengembangan aplikasi Android berbasis Jetpack Compose untuk mengelola multimedia seperti kamera, galeri, audio, dan video. Mahasiswa mempelajari pratinjau media, gesture interaktif (zoom dan geser), pemutaran audio–video, serta pengelolaan file multimedia secara aman dan responsif
Tools	:	Android Studio
Dosen Pengampu	:	**Muhayat, M.IT**
Durasi	:	1×120 menit

Tujuan Praktikum

1. Memahami konsep pengelolaan multimedia (audio, video, dan gambar) pada aplikasi Android.
2. Mengimplementasikan Camera & Gallery untuk mengambil dan memilih media menggunakan Jetpack Compose.
3. Menampilkan preview foto langsung di dalam aplikasi tanpa berpindah screen.
4. Menerapkan gesture interaktif pada media berupa:
   - ✓ pinch zoom (memperbesar dan memperkecil),
   - ✓ pan / geser untuk melihat detail media.
5. Menangani perubahan orientasi layar (portrait & landscape) tanpa kehilangan state media.
6. Menyimpan foto hasil kamera secara permanen ke Galeri perangkat menggunakan MediaStore (Scoped Storage).
7. Mengembangkan Audio Recorder untuk merekam suara dan menampilkan daftar file audio.
8. Mengimplementasikan Audio Player untuk memutar file audio hasil rekaman atau dari galeri.
9. Mengembangkan Video Player berbasis ExoPlayer dengan fitur:
   - ✓ pemutaran video,
   - ✓ fullscreen mode,
   - ✓ zoom dan pan pada tampilan video.
10. Mengelola file multimedia secara aman, meliputi:
    - ✓ pengambilan daftar file,
    - ✓ pemutaran media,
    - ✓ penghapusan dan pengubahan nama file.
11. Menerapkan arsitektur UI yang terstruktur dengan pemisahan screen, navigation, dan utility.
12. Memahami integrasi gesture handling dan multimedia playback dalam satu aplikasi Android modern.
13. Mengembangkan aplikasi multimedia yang responsif, interaktif, dan user-friendly.

Konsep Dasar

1. Konsep dasar multimedia pada Android (image, audio, dan video).
2. Penggunaan Camera dan Gallery API untuk mengambil dan memilih media.
3. Penerapan Jetpack Compose dalam pengembangan antarmuka multimedia.
4. Penanganan preview media langsung di dalam screen tanpa berpindah halaman.
5. Implementasi gesture interaktif pada media:
   - ✓ pinch zoom,
   - ✓ pan / geser.
6. Pengelolaan state UI pada Compose untuk media yang bersifat dinamis.
7. Penanganan perubahan orientasi layar (portrait dan landscape) secara aman.
8. Penyimpanan media menggunakan MediaStore (Scoped Storage).
9. Pengembangan Audio Recorder dan Audio Player pada Android.
10. Pemanfaatan ExoPlayer untuk pemutaran video.
11. Integrasi fullscreen mode pada video.
12. Pengelolaan file multimedia:
    - ✓ membaca daftar file,
    - ✓ memutar media,
    - ✓ menghapus dan mengubah nama file.
13. Penerapan arsitektur aplikasi terstruktur (UI, Navigation, Utility).
14. Penggunaan Android modern API dan best practice pengembangan aplikasi multimedia.
15. Peningkatan user experience (UX) melalui interaksi sentuh dan gesture.

Library & Tools Pendukung

- Tools Pengembangan
  - ✓ Android Studio
    Digunakan sebagai IDE utama untuk pengembangan, debugging, dan pengujian aplikasi Android.
  - ✓ Android SDK
    Menyediakan API Android untuk pengelolaan kamera, media, dan sistem file.
  - ✓ Android Emulator / Real Device
    Digunakan untuk pengujian fitur kamera, audio, dan video.
- Library Android & Jetpack
  - ✓ Jetpack Compose
    Digunakan untuk membangun antarmuka pengguna secara deklaratif.
  - ✓ Material 3 (Compose)
    Digunakan untuk komponen UI seperti Button, Card, Icon, dan TopAppBar.
  - ✓ Navigation Compose
    Digunakan untuk mengatur navigasi antar screen dalam aplikasi.

- Multimedia & Media Playback
  - ✓ MediaStore (Scoped Storage)
    Digunakan untuk menyimpan dan mengakses file multimedia secara aman.
  - ✓ AudioRecord / MediaRecorder
    Digunakan untuk merekam audio.
  - ✓ ExoPlayer (AndroidX Media3)
    Digunakan untuk pemutaran video dengan dukungan fullscreen dan kontrol lanjutan.
- Gesture & Interaksi
  - ✓ Pointer Input & Gesture API (Compose)
    Digunakan untuk menangani gesture pinch zoom dan pan pada image dan video.
- Bahasa & Build System
  - ✓ Kotlin
    Bahasa pemrograman utama dalam pengembangan aplikasi Android.
  - ✓ Gradle
    Digunakan untuk manajemen dependensi dan proses build aplikasi.

🗀 Struktur Minimal Project Praktikum #8

```
p8_multimedia_nimanda/
│
├── MainActivity.kt
├── ui/
│   │
│   ├── gallery/
│   │   └── CameraGalleryScreen.kt
│   ├── home/
│   │   └── HomeScreen.kt
│   ├── player/
│   │   └── AudioPlayerScreen.kt
│   ├── recorder/
│   │   └── AudioRecorderScreen.kt
│   ├── video/
│   │   └── VideoPlayerScreen.kt
│   ├── theme/
│   │   ├── Color.kt
│   │   ├── Theme.kt
│   │   └── Type.kt
│   ├── NavGraph.kt
│   └── Screens.kt
└── util/
    ├── FileManagerUtility.kt
    └── VideoFileData.kt
```

Rubrik Penilaian

No	Aspek Penilaian	Bobot (%)
1	Implementasi Camera & Gallery (take photo, pilih media)	15%
2	Image Preview (zoom, geser, responsif orientasi)	15%
3	Penyimpanan Foto ke Galeri (MediaStore)	10%
4	Audio Recorder & Audio Player	15%
5	Video Player (playback & fullscreen)	15%
6	Zoom & Pan pada Video Preview	10%
7	Manajemen File Multimedia (list, rename, delete)	10%
8	Struktur Proyek & Kerapian Kode	5%
	Total	100%

---------- Langkah-Langkah Praktikum ----------

Langkah 1 — Setup Proyek & Konfigurasi Dasar ----------

Target Belajar

1. Proyek Compose bisa run tanpa error dan siap dikembangkan.
2. Setup Kotlin 2.0.21, AGP 8.13.1, Compose Material3.
3. Tambah plugin org.jetbrains.kotlin.plugin.compose.
4. Perbaikan theme XML + warna + manifest + uses-feature kamera.
5. Build sukses dan Home Screen tampil.

Langkah Praktikum

1.1 Buka **Android Studio** → New Project

1. Pilih "Empty Activity"
2. Isi data:
   - Name: p8_multimedia_nimanda
   - Package name: id.antasari.p8_multimedia_nimanda
   - Language: Kotlin
   - Minimum SDK: API 24 (Android 7.0)
3. Klik **Finish**

1.2 Update File : **settings.gradle.kts**

```
pluginManagement {
    repositories {
        gradlePluginPortal()
        google()
        mavenCentral()
    }
    plugins {
        // AGP dan Kotlin plugin versions
        id("com.android.application") version "8.13.1"
        id("com.android.library") version "8.13.1"
        id("org.jetbrains.kotlin.android") version "2.0.21"
    }
}

dependencyResolutionManagement {
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
    }
}

rootProject.name = "P8Multimedia"
include(":app")
```

1.3 Update File : **gradle.properties**

```
# Gradle / Kotlin properties
kotlin.code.style=official
android.useAndroidX=true

# Optional: customize JVM target
org.gradle.jvmargs=-Xmx2048m
```

1.4 Update File : **build.gradle.kts (root)**

```
// build.gradle.kts (root)
plugins {
    // Tidak menambahkan plugin yang memerlukan repositories di sini
}

// contoh task global
tasks.register("clean", Delete::class) {
    delete(rootProject.buildDir)
}
```

1.5 Update File : **app/build.gradle.kts**

```
plugins {
    id("com.android.application")
    id("org.jetbrains.kotlin.android")
    id("org.jetbrains.kotlin.plugin.compose") version "2.0.21"
}

android {
    namespace = "id.antasari.p8_multimedia_nimanda"
    compileSdk = 34  // tetap 34 agar bisa compile API 34

    defaultConfig {
        applicationId = "id.antasari.p8_multimedia_nimanda"
        minSdk = 24
        targetSdk = 33   // <-- DITURUNKAN SEMENTARA KE 33
        versionCode = 1
        versionName = "1.0"

        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }

    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_17
        targetCompatibility = JavaVersion.VERSION_17
    }
    kotlinOptions {
        jvmTarget = "17"
    }

    buildFeatures {
        compose = true
    }
```

```
    packagingOptions {
        resources {
            excludes += "/META-INF/{AL2.0,LGPL2.1}"
        }
    }

    buildTypes {
        release {
            isMinifyEnabled = false
        }
    }
}

dependencies {
    val composeBom = platform("androidx.compose:compose-bom:2024.10.00")
    implementation(composeBom)
    implementation("androidx.compose.material3:material3")

    implementation("androidx.compose.ui:ui")
    implementation("androidx.compose.material3:material3")
    implementation("androidx.activity:activity-compose:1.9.0")
    implementation("androidx.navigation:navigation-compose:2.7.0")
    implementation("androidx.lifecycle:lifecycle-runtime-ktx:2.6.2")
    implementation("androidx.lifecycle:lifecycle-viewmodel-compose:2.6.2")
    implementation("io.coil-kt:coil-compose:2.4.0")
    implementation("androidx.compose.material:material-icons-core")
    implementation("androidx.compose.material:material-icons-extended")
    implementation("androidx.compose.material3:material3:1.1.0")
    implementation("com.google.android.material:material:1.9.0")

    testImplementation("junit:junit:4.13.2")
    androidTestImplementation("androidx.test.ext:junit:1.1.6")
    androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
    androidTestImplementation("androidx.compose.ui:ui-test-junit4")
}
```

1.6 Update File : app/src/main/**AndroidManifest.xml**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.antasari.p8_multimedia_nimanda">

    <!-- Permissions -->
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.CAMERA" />
    <!-- Perhatikan: untuk Android 13+ akan gunakan READ_MEDIA_* saat implementasi runtime permission -->

    <!-- Jelaskan apakah app membutuhkan hardware camera atau tidak.
         required="false" → app tetap bisa terinstall di device tanpa camera -->
    <uses-feature android:name="android.hardware.camera" android:required="false" />
```

```
    <application
        android:allowBackup="true"
        android:label="Multimedia Studio"
        android:icon="@mipmap/ic_launcher"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:theme="@style/Theme.P8Multimedia">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

1.7 Update File : /ui/theme/**Color.kt**

```
package id.antasari.p8_multimedia_nimanda.ui.theme

import androidx.compose.ui.graphics.Color

val GreenLight = Color(0xFF2ECC71)
val GreenPrimary = Color(0xFF00C06B)
val GreenDark = Color(0xFF007A4D)
val Background = Color(0xFFF5F7F6)
```

1.8 Update File : /ui/theme/**Theme.kt**

```
package id.antasari.p8_multimedia_nimanda.ui.theme

import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.graphics.Color

private val LightColors = lightColorScheme(
    primary = GreenPrimary,
    onPrimary = Color.White,
    secondary = GreenLight,
    background = Background,
    surface = Color.White,
    onBackground = Color.Black,
    onSurface = Color.Black
)

@Composable
fun P8MultimediaTheme(content: @Composable () -> Unit) {
    MaterialTheme(
        colorScheme = LightColors,
        typography = Typography(),
        shapes = Shapes(),
        content = content
    )
}
```

1.9 Update File : app/src/main/res/values/**strings.xml**

```
<resources>
    <string name="app_name">MMStudio</string>
</resources>
```

1.10 Update File : app/src/main/res/values/**themes.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<resources xmlns:tools="http://schemas.android.com/tools">

    <!-- Theme yang dipakai Activity (menurunkan Material3 DayNight) -->
    <style name="Theme.P8Multimedia" parent="Theme.Material3.DayNight.NoActionBar">
        <!-- Sesuaikan warna dasar/warna status bar jika mau -->
        <!-- Gunakan colorPrimary/colorSecondary atau colorScheme jika diperlukan -->
        <item name="android:statusBarColor">@color/green_primary</item>
        <item name="android:navigationBarColor">@color/white</item>
    </style>

</resources>
```

1.11 Update File : app/src/main/res/values/**colors.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="green_light">#2ECC71</color>
    <color name="green_primary">#00C06B</color>
    <color name="green_dark">#007A4D</color>
    <color name="white">#FFFFFF</color>
    <color name="background">#F5F7F6</color>
</resources>
```

1.12 Update File : **MainActivity**.kt

```
package id.antasari.p8_multimedia_nimanda

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.CameraAlt
import androidx.compose.material.icons.filled.Mic
import androidx.compose.material.icons.filled.PlayArrow
import androidx.compose.material.icons.filled.Videocam
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
```

```
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import id.antasari.p8_multimedia_nimanda.ui.theme.P8MultimediaTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            P8MultimediaTheme {
                Surface(modifier = Modifier.fillMaxSize()) {
                    HomeScreen()
                }
            }
        }
    }
}

@Composable
fun HomeScreen() {
    // Warna gradient sesuai mock-up (hijau ke hijau lebih gelap)
    val topGradient = Brush.verticalGradient(
        colors = listOf(Color(0xFF2ECC71), Color(0xFF00A86B))
    )
    Column(
        modifier = Modifier
            .fillMaxSize()
            .background(color = Color(0xFFF5F7F6))
    ) {
        // Header area with gradient and round icon
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(240.dp)
                .background(brush = topGradient),
            contentAlignment = Alignment.TopCenter
        ) {
            Column(
                horizontalAlignment = Alignment.CenterHorizontally,
                modifier = Modifier.fillMaxWidth()
            ) {
                Spacer(modifier = Modifier.height(28.dp))
                // Circle icon
                Box(
                    modifier = Modifier
                        .size(86.dp)
                        .clip(CircleShape)
                        .background(Color(0xFF00C06B)),
                    contentAlignment = Alignment.Center
                ) {
```

```
            // Icon inside circle (play)
            Icon(
                imageVector = Icons.Default.PlayArrow,
                contentDescription = "Logo",
                tint = Color.White,
                modifier = Modifier.size(44.dp)
            )
        }
        Spacer(modifier = Modifier.height(12.dp))
        Text(
            text = "Multimedia Studio",
            fontSize = 22.sp,
            fontWeight = FontWeight.SemiBold,
            color = Color.White
        )
    }
}

// Content card area
Column(
    modifier = Modifier
        .fillMaxSize()
        .padding(horizontal = 18.dp)
        .offset(y = (-36).dp) // overlap effect like mock-up
) {
            // Grid 2x2 buttons
            Column(
                modifier = Modifier
                    .fillMaxWidth()
            ) {
                Row(
                    modifier = Modifier.fillMaxWidth(),
                    horizontalArrangement = Arrangement.spacedBy(14.dp)
                ) {
                    MenuCard(
                        icon = Icons.Default.Mic,
                        title = "Record Audio",
                        modifier = Modifier.weight(1f)
                    )
                    MenuCard(
                        icon = Icons.Default.PlayArrow,
                        title = "Play Audio",
                        modifier = Modifier.weight(1f)
                    )
                }
                Spacer(modifier = Modifier.height(14.dp))
                Row(
                    modifier = Modifier.fillMaxWidth(),
                    horizontalArrangement = Arrangement.spacedBy(14.dp)
                ) {
```

```
                MenuCard(
                    icon = Icons.Default.Videocam,
                    title = "Play Video",
                    modifier = Modifier.weight(1f)
                )
                MenuCard(
                    icon = Icons.Default.CameraAlt,
                    title = "Camera & Gallery",
                    modifier = Modifier.weight(1f)
                )
            }
        }

        Spacer(modifier = Modifier.weight(1f))

        // Footer mirrored from mock-up
        Column(
            modifier = Modifier
                .fillMaxWidth()
                .padding(bottom = 18.dp),
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
            Text(
                text = "Copyright ©2025",
                style = MaterialTheme.typography.bodySmall
            )
            Text(
                text = "Praktikum #8 Menggunakan Multimedia",
                style = MaterialTheme.typography.bodySmall
            )
            Text(
                text = "Kuliah Mobile Programming S1 Teknologi Informasi",
                style = MaterialTheme.typography.bodySmall
            )
            Text(
                text = "UIN Antasari Banjarmasin",
                style = MaterialTheme.typography.bodySmall
            )
        }
    }
}
}

@Composable
fun MenuCard(icon: androidx.compose.ui.graphics.vector.ImageVector, title: String, modifier: Modifier = Modifier) {
    Card(
        modifier = modifier
            .height(120.dp),
        shape = RoundedCornerShape(16.dp),
        elevation = CardDefaults.cardElevation(defaultElevation = 6.dp)
    ) {
```

```
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(12.dp),
            verticalArrangement = Arrangement.Center,
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
            Box(
                modifier = Modifier
                    .size(54.dp)
                    .clip(CircleShape)
                    .background(Color(0xFF00C06B)),
                contentAlignment = Alignment.Center
            ) {
                Icon(
                    imageVector = icon,
                    contentDescription = title,
                    tint = Color.White,
                    modifier = Modifier.size(28.dp)
                )
            }
            Spacer(modifier = Modifier.height(10.dp))
            Text(text = title, style = MaterialTheme.typography.bodyMedium)
        }
    }
}
```

1.13 Run & Checklist

- ✓ Simpan file.
- ✓ Di Android Studio → File → Sync Project with Gradle Files.
- ✓ Kemudian lakukan: Build → Rebuild Project.
- ✓ Run app di emulator / device.

Checklist

- ☐ app/build.gradle.kts sudah berisi implementation("com.google.android.material:material:1.9.0").
- ☐ res/values/themes.xml memakai parent tema yang tersedia (Theme.Material3.DayNight.NoActionBar atau fallback Theme.MaterialComponents.DayNight.NoActionBar).
- ☐ Gradle sync sukses tanpa error.
- ☐ App dapat dijalankan dan Activity memakai tema baru tanpa crash.

Langkah 2 — Setup Navigation & Struktur Folder ----------

Target Belajar

1. Struktur folder proyek yang rapi untuk fitur UI (ui/home, ui/recorder, ui/player, ui/video, ui/gallery),
2. NavHost Compose terpusat yang mengatur 5 route: home, audio_recorder, audio_player, video_player, camera_gallery,
3. MainActivity sudah menggunakan NavHost, dan setiap screen punya TopBar (back navigation untuk screen selain Home) serta placeholder UI yang siap dikembangkan lebih lanjut.

Langkah Praktikum

2.1 Buat **Struktur Folder** & **File** Project berikut ini :

```
p8_multimedia_nimanda/
│
├── MainActivity.kt
├── ui/
│   │
│   ├── gallery/
│   │   └── CameraGalleryScreen.kt
│   ├── home/
│   │   └── HomeScreen.kt
│   ├── player/
│   │   └── AudioPlayerScreen.kt
│   ├── recorder/
│   │   └── AudioRecorderScreen.kt
│   ├── video/
│   │   └── VideoPlayerScreen.kt
│   ├── theme/
│   │   ├── Color.kt
│   │   ├── Theme.kt
│   │   └── Type.kt
│   ├── NavGraph.kt
│   └── Screens.kt
└── util/
    ├── FileManagerUtility.kt
    └── VideoFileData.kt
```

2.2 Update file : **Screens.kt** — definisi route

```
package id.antasari.p8_multimedia_nimanda.ui

sealed class Screen(val route: String, val title: String) {
    object Home : Screen("home", "Multimedia Studio")
    object AudioRecorder : Screen("audio_recorder", "Audio Recorder")
    object AudioPlayer : Screen("audio_player", "Audio Player")
    object VideoPlayer : Screen("video_player", "Video Player")
    object CameraGallery : Screen("camera_gallery", "Camera & Gallery")
}
```

2.3 Update file : **NavGraph.kt** — NavHost & route registration

```
package id.antasari.p8_multimedia_nimanda.ui

import androidx.compose.runtime.Composable
import androidx.navigation.NavHostController
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import androidx.navigation.compose.rememberNavController
import id.antasari.p8_multimedia_nimanda.ui.home.HomeScreen
```

```
import id.antasari.p8_multimedia_nimanda.ui.recorder.AudioRecorderScreen
import id.antasari.p8_multimedia_nimanda.ui.player.AudioPlayerScreen
import id.antasari.p8_multimedia_nimanda.ui.video.VideoPlayerScreen
import id.antasari.p8_multimedia_nimanda.ui.gallery.CameraGalleryScreen

@Composable
fun AppNavHost(startDestination: String = Screen.Home.route) {
    val navController = rememberNavController()
    NavHost(navController = navController, startDestination = startDestination) {
        composable(Screen.Home.route) {
            HomeScreen(onNavigate = { route -> navController.navigate(route) })
        }
        composable(Screen.AudioRecorder.route) {
            AudioRecorderScreen(onBack = { navController.popBackStack() })
        }
        composable(Screen.AudioPlayer.route) {
            AudioPlayerScreen(onBack = { navController.popBackStack() })
        }
        composable(Screen.VideoPlayer.route) {
            VideoPlayerScreen(onBack = { navController.popBackStack() })
        }
        composable(Screen.CameraGallery.route) {
            CameraGalleryScreen(onBack = { navController.popBackStack() })
        }
    }
}
```

2.4 Update file : **MainActivity.kt** — entrypoint

```
package id.antasari.p8_multimedia_nimanda

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import id.antasari.p8_multimedia_nimanda.ui.AppNavHost
import id.antasari.p8_multimedia_nimanda.ui.theme.P8MultimediaTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            P8MultimediaTheme {
                AppNavHost()
            }
        }
    }
}
```

2.5 Update file : ui/home/**HomeScreen.kt** — Home dengan navigasi dari kartu

```
package id.antasari.p8_multimedia_nimanda.ui.home

import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.CameraAlt
import androidx.compose.material.icons.filled.Mic
import androidx.compose.material.icons.filled.PlayArrow
import androidx.compose.material.icons.filled.Videocam
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.unit.dp
import id.antasari.p8_multimedia_nimanda.ui.Screen

@Composable
fun HomeScreen(onNavigate: (String) -> Unit) {
    val topGradient = Brush.verticalGradient(colors = listOf(Color(0xFF2ECC71), Color(0xFF00A86B)))
    Column(
        modifier = Modifier
            .fillMaxSize()
            .background(color = Color(0xFFF5F7F6))
    ) {
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(220.dp)
                .background(brush = topGradient),
            contentAlignment = Alignment.TopCenter
        ) {
            Column(horizontalAlignment = Alignment.CenterHorizontally) {
                Spacer(Modifier.height(28.dp))
                Box(
                    modifier = Modifier
                        .size(86.dp)
                        .clip(CircleShape)
                        .background(Color(0xFF00C06B)),
                    contentAlignment = Alignment.Center
                ) {
                    Icon(
                        imageVector = Icons.Default.PlayArrow,
                        contentDescription = "Logo",
                        tint = Color.White,
                        modifier = Modifier.size(44.dp)
                    )
                }
                Spacer(Modifier.height(12.dp))
                Text("Multimedia Studio", style = MaterialTheme.typography.titleMedium, color = Color.White)
            }
        }
```

```
        Column(modifier = Modifier
            .fillMaxSize()
            .padding(horizontal = 18.dp)
            .offset(y = (-36).dp)
        ) {
            // Cards 2x2
            Row(modifier = Modifier.fillMaxWidth(), horizontalArrangement = Arrangement.spacedBy(14.dp)) {
                MenuCard(icon = Icons.Default.Mic, title = "Record Audio", onClick = { onNavigate(Screen.AudioRecorder.route) }, modifier = Modifier.weight(1f))
                MenuCard(icon = Icons.Default.PlayArrow, title = "Play Audio", onClick = { onNavigate(Screen.AudioPlayer.route) }, modifier = Modifier.weight(1f))
            }
            Spacer(Modifier.height(14.dp))
            Row(modifier = Modifier.fillMaxWidth(), horizontalArrangement = Arrangement.spacedBy(14.dp)) {
                MenuCard(icon = Icons.Default.Videocam, title = "Play Video", onClick = { onNavigate(Screen.VideoPlayer.route) }, modifier = Modifier.weight(1f))
                MenuCard(icon = Icons.Default.CameraAlt, title = "Camera & Gallery", onClick = { onNavigate(Screen.CameraGallery.route) }, modifier = Modifier.weight(1f))
            }

            Spacer(modifier = Modifier.weight(1f))

            Column(modifier = Modifier.fillMaxWidth().padding(bottom = 18.dp), horizontalAlignment = Alignment.CenterHorizontally) {
                Text("Copyright ©2025", style = MaterialTheme.typography.bodySmall)
                Text("Praktikum #8 Menggunakan Multimedia", style = MaterialTheme.typography.bodySmall)
                Text("Kuliah Mobile Programming S1 Teknologi Informasi", style = MaterialTheme.typography.bodySmall)
                Text("UIN Antasari Banjarmasin", style = MaterialTheme.typography.bodySmall)
            }
        }
    }
}

@Composable
fun MenuCard(icon: androidx.compose.ui.graphics.vector.ImageVector, title: String, modifier: Modifier = Modifier, onClick: () -> Unit) {
    Card(modifier = modifier
        .height(120.dp)
        .padding(vertical = 4.dp),
        shape = RoundedCornerShape(16.dp),
        elevation = CardDefaults.cardElevation(defaultElevation = 6.dp),
        onClick = onClick
    ) {
        Column(modifier = Modifier.fillMaxSize().padding(12.dp), verticalArrangement = Arrangement.Center, horizontalAlignment = Alignment.CenterHorizontally) {
            Box(modifier = Modifier.size(54.dp).clip(CircleShape).background(Color(0xFF00C06B)), contentAlignment = Alignment.Center) {
                Icon(imageVector = icon, contentDescription = title, tint = Color.White, modifier = Modifier.size(28.dp))
            }
            Spacer(Modifier.height(10.dp))
            Text(title, style = MaterialTheme.typography.bodyMedium)
        }
    }
}
```

2.6 Update file : ui/recorder/**AudioRecorderScreen.kt** — placeholder & TopBar

```
package id.antasari.p8_multimedia_nimanda.ui.recorder

import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.material.icons.filled.Mic
import androidx.compose.material3.Button
import androidx.compose.material3.FloatingActionButton
import androidx.compose.material3.Icon
import androidx.compose.material3.IconButton
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.vector.ImageVector
import androidx.compose.foundation.background
import androidx.compose.foundation.clickable
import androidx.compose.ui.graphics.Color
```

```
@Composable
fun AudioRecorderScreen(onBack: () -> Unit) {
    Column(modifier = Modifier.fillMaxSize()) {
        AppTopBar(title = "Audio Recorder", navigationIcon = Icons.Default.ArrowBack, onNavigateUp = onBack)

        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(horizontal = 24.dp),
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
            Spacer(modifier = Modifier.height(24.dp))
            Text(text = "Ready to record", style = MaterialTheme.typography.bodyMedium)
            Spacer(modifier = Modifier.height(20.dp))

            // Big mic circle area
            Box(
                modifier = Modifier
                    .size(160.dp),
                contentAlignment = Alignment.Center
            ) {
                Surface(
                    shape = CircleShape,
                    tonalElevation = 4.dp,
                    modifier = Modifier.size(140.dp)
                ) {
                    Box(contentAlignment = Alignment.Center) {
                        FloatingActionButton(
                            onClick = { /* TODO: record action */ },
                            shape = CircleShape,
                            modifier = Modifier.size(72.dp)
                        ) {
                            Icon(
                                imageVector = Icons.Default.Mic,
                                contentDescription = "Record"
                            )
                        }
                    }
                }
            }

            Spacer(modifier = Modifier.height(28.dp))

            Button(
                onClick = { /* TODO: start/stop recording logic */ },
                modifier = Modifier
                    .fillMaxWidth()
                    .height(48.dp)
            ) {
                Text(text = "Start Recording")
            }
        }
    }
}
```

```
@Composable
fun AppTopBar(title: String, navigationIcon: ImageVector? = null, onNavigateUp: () -> Unit = {}) {
    Surface(
        tonalElevation = 2.dp,
        shadowElevation = 2.dp,
        modifier = Modifier
            .fillMaxWidth()
            .background(color = MaterialTheme.colorScheme.surface)
    ) {
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .height(56.dp)
                .padding(horizontal = 8.dp),
            verticalAlignment = Alignment.CenterVertically
        ) {
            if (navigationIcon != null) {
                IconButton(onClick = onNavigateUp) {
                    Icon(imageVector = navigationIcon, contentDescription = "Back")
                }
            } else {
                Spacer(modifier = Modifier.width(48.dp))
            }

            Spacer(modifier = Modifier.width(8.dp))

            Text(
                text = title,
                style = MaterialTheme.typography.titleMedium,
                color = MaterialTheme.colorScheme.onSurface
            )
        }
    }
}
```

2.7 Update file : ui/player/**AudioPlayerScreen.kt** — placeholder

```
package id.antasari.p8_multimedia_nimanda.ui.player

import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.material.icons.filled.Pause
import androidx.compose.material.icons.filled.PlayArrow
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.foundation.background
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.draw.clip
```

```
@Composable
fun AudioPlayerScreen(onBack: () -> Unit) {
    Column(modifier = Modifier.fillMaxSize()) {
        AppTopBar(title = "Audio Player", navigationIcon = Icons.Default.ArrowBack, onNavigateUp = onBack)

        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(horizontal = 20.dp),
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
            Spacer(modifier = Modifier.height(20.dp))

            // cover card
            Card(
                modifier = Modifier
                    .size(width = 220.dp, height = 220.dp),
                shape = RoundedCornerShape(16.dp),
                elevation = CardDefaults.cardElevation(defaultElevation = 8.dp)
            ) {
                Box(
                    modifier = Modifier
                        .fillMaxSize()
                        .background(MaterialTheme.colorScheme.primary),
                    contentAlignment = Alignment.Center
                ) {
                    // placeholder circle
                    Box(
                        modifier = Modifier
                            .size(72.dp)
                            .clip(RoundedCornerShape(36.dp))
                            .background(Color.White.copy(alpha = 0.12f)),
                        contentAlignment = Alignment.Center
                    ) {
                        Icon(
                            imageVector = Icons.Default.PlayArrow,
                            contentDescription = "Play",
                            tint = Color.White,
                            modifier = Modifier.size(36.dp)
                        )
                    }
                }
            }

            Spacer(modifier = Modifier.height(18.dp))

Text("Nature Sounds", style = MaterialTheme.typography.titleMedium)
Text("Ambient Recording", style = MaterialTheme.typography.bodySmall, color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.6f))

Spacer(modifier = Modifier.height(18.dp))
```

```
            // slider placeholder
            Column(modifier = Modifier.fillMaxWidth()) {
                Slider(
                    value = 0.25f,
                    onValueChange = { /* TODO: sync with player position */ },
                    modifier = Modifier
                        .fillMaxWidth()
                )
                Row(modifier = Modifier.fillMaxWidth(), horizontalArrangement = Arrangement.SpaceBetween) {
                    Text("1:25", style = MaterialTheme.typography.bodySmall)
                    Text("4:03", style = MaterialTheme.typography.bodySmall)
                }
            }

            Spacer(modifier = Modifier.height(18.dp))

            // controls
            Row(verticalAlignment = Alignment.CenterVertically, horizontalArrangement = Arrangement.spacedBy(24.dp)) {
                IconButton(onClick = { /* prev */ }) {
                    Icon(Icons.Default.PlayArrow, contentDescription = "Prev")
                }

                // play/pause circular button
                FloatingActionButton(onClick = { /* play/pause */ }) {
                    Icon(Icons.Default.PlayArrow, contentDescription = "Play")
                }

                IconButton(onClick = { /* next */ }) {
                    Icon(Icons.Default.PlayArrow, contentDescription = "Next")
                }
            }
        }
    }
}
@Composable
fun AppTopBar(title: String, navigationIcon: androidx.compose.ui.graphics.vector.ImageVector? = null, onNavigateUp: () -> Unit = {}) {
    Surface(
        tonalElevation = 2.dp,
        modifier = Modifier
            .fillMaxWidth()
            .background(color = MaterialTheme.colorScheme.surface)
    ) {
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .height(56.dp)
                .padding(horizontal = 8.dp),
            verticalAlignment = Alignment.CenterVertically
        ) {
            if (navigationIcon != null) {
                IconButton(onClick = onNavigateUp) {
                    Icon(imageVector = navigationIcon, contentDescription = "Back")
                }
            } else {
                Spacer(modifier = Modifier.width(48.dp))
            }

          Spacer(modifier = Modifier.width(8.dp))

          Text(
              text = title,
              style = MaterialTheme.typography.titleMedium,
              color = MaterialTheme.colorScheme.onSurface
          )
        }
    }
}
```

2.8 Update file : ui/video/**VideoPlayerScreen.kt** — placeholder

```
package id.antasari.p8_multimedia_nimanda.ui.video

import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.material.icons.filled.PlayArrow
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.foundation.background
import androidx.compose.ui.graphics.Color

@Composable
fun VideoPlayerScreen(onBack: () -> Unit) {
    Column(modifier = Modifier.fillMaxSize()) {
        AppTopBar(title = "Video Player", navigationIcon = Icons.Default.ArrowBack, onNavigateUp = onBack)

        Column(modifier = Modifier
            .fillMaxSize()
            .padding(horizontal = 16.dp)) {
            Spacer(modifier = Modifier.height(16.dp))

            // video card / preview
            Card(
                modifier = Modifier
                    .fillMaxWidth()
                    .height(160.dp),
                shape = RoundedCornerShape(16.dp),
                elevation = CardDefaults.cardElevation(defaultElevation = 8.dp)
            ) {
               Box(modifier = Modifier.fillMaxSize(), contentAlignment = Alignment.Center) {
                   // placeholder: play icon + thin progress line
                   Column(horizontalAlignment = Alignment.CenterHorizontally) {
                       Icon(Icons.Default.PlayArrow, contentDescription = "Play", modifier = Modifier.size(36.dp))
                       Spacer(modifier = Modifier.height(8.dp))
                       // fake progress line
                       Box(modifier = Modifier
                           .width(240.dp)
                           .height(4.dp)
                           .background(Color.LightGray.copy(alpha = 0.4f))
                       )
                       Spacer(modifier = Modifier.height(6.dp))
                       Text("1:17 / 3:05", style = MaterialTheme.typography.bodySmall)
                   }
               }
            }

            Spacer(modifier = Modifier.height(14.dp))
```

```
            // details card
            Card(
                modifier = Modifier
                    .fillMaxWidth(),
                shape = RoundedCornerShape(12.dp),
                elevation = CardDefaults.cardElevation(defaultElevation = 4.dp)
            ) {
                Column(modifier = Modifier.padding(16.dp)) {
                    Text("Nature Documentary", style = MaterialTheme.typography.titleMedium)
                    Spacer(modifier = Modifier.height(8.dp))
                    Text("A beautiful journey through natural landscapes and wildlife habitats.", style = MaterialTheme.typography.bodySmall)
                    Spacer(modifier = Modifier.height(12.dp))
                    Row(modifier = Modifier.fillMaxWidth(), horizontalArrangement = Arrangement.SpaceBetween) {
                        Column {
                            Text("Duration", style = MaterialTheme.typography.labelSmall)
                            Text("3:05", style = MaterialTheme.typography.bodySmall)
                        }
                        Column {
                            Text("Resolution", style = MaterialTheme.typography.labelSmall)
                            Text("1920 x 1080", style = MaterialTheme.typography.bodySmall)
                        }
                        Column {
                            Text("Format", style = MaterialTheme.typography.labelSmall)
                            Text("MP4", style = MaterialTheme.typography.bodySmall)
                        }
                    }
                }
            }
        }
    }
}
@Composable
fun AppTopBar(title: String, navigationIcon: androidx.compose.ui.graphics.vector.ImageVector? = null, onNavigateUp: () -> Unit = {}) {
    Surface(
        tonalElevation = 2.dp,
        modifier = Modifier
            .fillMaxWidth()
            .background(color = MaterialTheme.colorScheme.surface)
    ) {
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .height(56.dp)
                .padding(horizontal = 8.dp),
            verticalAlignment = Alignment.CenterVertically
        ) {
            if (navigationIcon != null) {
                IconButton(onClick = onNavigateUp) {
                    Icon(imageVector = navigationIcon, contentDescription = "Back")
                }
            } else {
                Spacer(modifier = Modifier.width(48.dp))
            }

            Spacer(modifier = Modifier.width(8.dp))

            Text(
                text = title,
                style = MaterialTheme.typography.titleMedium,
                color = MaterialTheme.colorScheme.onSurface
            )
        }
    }
}
```

2.9 Update file : ui/gallery/**CameraGalleryScreen.kt** — placeholder

```
package id.antasari.p8_multimedia_nimanda.ui.gallery

import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.material.icons.filled.CameraAlt
import androidx.compose.material.icons.filled.Image
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.ui.graphics.Color
import androidx.compose.foundation.clickable

@Composable
fun CameraGalleryScreen(onBack: () -> Unit) {
    Column(modifier = Modifier.fillMaxSize()) {
        AppTopBar(title = "Camera & Gallery", navigationIcon = Icons.Default.ArrowBack, onNavigateUp = onBack)

        Column(modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)) {
            // Open Camera button (green)
            Card(
                modifier = Modifier
                    .fillMaxWidth()
                    .height(64.dp),
                shape = RoundedCornerShape(12.dp)
            ) {
                Row(
                    modifier = Modifier
                        .fillMaxSize()
                        .padding(horizontal = 12.dp),
                    verticalAlignment = Alignment.CenterVertically
                ) {
                    IconButton(onClick = { /* TODO: open camera */ }) {
                        Icon(Icons.Default.CameraAlt, contentDescription = "Open Camera")
                    }
                    Column {
                        Text("Open Camera", style = MaterialTheme.typography.titleMedium)
                        Text("Take a new photo", style = MaterialTheme.typography.bodySmall, color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.6f))
                    }
                }
            }

            Spacer(modifier = Modifier.height(12.dp))

            // Choose from Gallery
            Card(
                modifier = Modifier
                    .fillMaxWidth()
                    .height(64.dp),
                shape = RoundedCornerShape(12.dp)
            ) {
                Row(
                    modifier = Modifier
                        .fillMaxSize()
                        .padding(horizontal = 12.dp),
                    verticalAlignment = Alignment.CenterVertically
                ) {
```

```
                IconButton(onClick = { /* TODO: pick gallery */ }) {
                    Icon(Icons.Default.Image, contentDescription = "Choose from Gallery")
                }
                Column {
                    Text("Choose from Gallery", style = MaterialTheme.typography.titleMedium)
                    Text("Select existing photo", style = MaterialTheme.typography.bodySmall, color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.6f))
                }
            }
        }

        Spacer(modifier = Modifier.height(18.dp))

        // Preview area
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(180.dp)
                .background(color = MaterialTheme.colorScheme.surface, shape = RoundedCornerShape(12.dp)),
            contentAlignment = Alignment.Center
        ) {
            Column(horizontalAlignment = Alignment.CenterHorizontally) {
                Icon(Icons.Default.Image, contentDescription = "No image", modifier = Modifier.size(48.dp), tint = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.6f))
                Spacer(modifier = Modifier.height(8.dp))
                Text("No image selected", style = MaterialTheme.typography.bodyMedium)
                Text("Choose an option above to get started", style = MaterialTheme.typography.bodySmall, color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.6f))
            }
        }
    }
}

/** Simple AppTopBar (self-contained) */
@Composable
fun AppTopBar(title: String, navigationIcon: androidx.compose.ui.graphics.vector.ImageVector? = null, onNavigateUp: () -> Unit = {}) {
    Surface(
        tonalElevation = 2.dp,
        modifier = Modifier
            .fillMaxWidth()
            .background(color = MaterialTheme.colorScheme.surface)
    ) {
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .height(56.dp)
                .padding(horizontal = 8.dp),
            verticalAlignment = Alignment.CenterVertically
        ) {
            if (navigationIcon != null) {
                IconButton(onClick = onNavigateUp) {
                    Icon(imageVector = navigationIcon, contentDescription = "Back")
                }
            } else {
                Spacer(modifier = Modifier.width(48.dp))
            }

            Spacer(modifier = Modifier.width(8.dp))

            Text(
                text = title,
                style = MaterialTheme.typography.titleMedium,
                color = MaterialTheme.colorScheme.onSurface
            )
        }
    }
}
```

2.10 Jalankan **Build** lalu **Run app** di emulator / device.

✓ Checklist UI & Kode

- ☐ Semua file Kotlin disimpan sesuai struktur folder.
- ☐ Saat app dijalankan: Home screen muncul.
- ☐ Ketika men-tap Record Audio → navigasi ke Audio Recorder screen dan tombol Back di TopBar kembali ke Home.
- ☐ Ketika men-tap Play Audio → navigasi ke Audio Player screen.
- ☐ Ketika men-tap Play Video → navigasi ke Video Player screen.
- ☐ Ketika men-tap Camera & Gallery → navigasi ke Camera & Gallery screen.

Langkah 3 — Implementasi UI Home ----------

Target Belajar

1. UI Home tampil dengan (header gradient, icon bundar, hero card, grid menu 2×2, footer).
2. Layout tetap rapi dan tidak overflow saat rotasi landscape.
3. User dapat scroll secara vertikal bila konten tidak muat.
4. Navigasi ke screen lain tetap berjalan (onNavigate()).

Langkah Praktikum

3.1 Update file : ui/home/**HomeScreen.kt**

```
package id.antasari.p8_multimedia_nimanda.ui.home

import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.CameraAlt
import androidx.compose.material.icons.filled.Mic
import androidx.compose.material.icons.filled.PlayArrow
import androidx.compose.material.icons.filled.Videocam
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import id.antasari.p8_multimedia_nimanda.R
import id.antasari.p8_multimedia_nimanda.ui.Screen
```

```
@Composable
fun HomeScreen(onNavigate: (String) -> Unit) {
    val scrollState = rememberScrollState()
    val gradient = Brush.verticalGradient(
        listOf(Color(0xFF2ECC71), Color(0xFF00A86B))
    )

    Column(
        modifier = Modifier
            .fillMaxSize()
            .background(Color(0xFFF5F7F6))
            .verticalScroll(scrollState)
    ) {

        // ================ HEADER ================
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(200.dp)
                .background(gradient),
            contentAlignment = Alignment.TopCenter
        ) {
            Column(horizontalAlignment = Alignment.CenterHorizontally) {
                Spacer(Modifier.height(18.dp))

                // LOGO IMAGE
                Image(
                    painter = painterResource(id = R.drawable.logo_multimedia),
                    contentDescription = "App Logo",
                    modifier = Modifier
                        .size(96.dp)
                        .clip(CircleShape),
                    contentScale = ContentScale.Crop
                )
                Spacer(Modifier.height(12.dp))

                Text(
                    text = "Multimedia Studio",
                    style = MaterialTheme.typography.titleLarge,
                    color = Color.White
                )
                Spacer(Modifier.height(24.dp))
            }
        }

        // ================ HERO CARD ================
        HeroCard(
            modifier = Modifier
                .padding(horizontal = 18.dp)
                .height(180.dp)
                .offset(y = (-26).dp)
        )
        Spacer(Modifier.height(0.dp))

        // ================ MENU GRID WITH OFFSET ================
        Column(
            modifier = Modifier
                .padding(horizontal = 18.dp)
        ) {
```

```
Row(
    modifier = Modifier.fillMaxWidth(),
    horizontalArrangement = Arrangement.spacedBy(14.dp)
) {
    MenuCard(
        icon = Icons.Default.Mic,
        title = "Record Audio",
        modifier = Modifier.weight(1f),
        onClick = { onNavigate(Screen.AudioRecorder.route) }
    )
    MenuCard(
        icon = Icons.Default.PlayArrow,
        title = "Play Audio",
        modifier = Modifier.weight(1f),
        onClick = { onNavigate(Screen.AudioPlayer.route) }
    )
}
Spacer(Modifier.height(14.dp))

Row(
    modifier = Modifier.fillMaxWidth(),
    horizontalArrangement = Arrangement.spacedBy(14.dp)
) {
    MenuCard(
        icon = Icons.Default.Videocam,
        title = "Play Video",
        modifier = Modifier.weight(1f),
        onClick = { onNavigate(Screen.VideoPlayer.route) }
    )
    MenuCard(
        icon = Icons.Default.CameraAlt,
        title = "Camera & Gallery",
        modifier = Modifier.weight(1f),
        onClick = { onNavigate(Screen.CameraGallery.route) }
    )
}
Spacer(Modifier.height(24.dp))

// ================ FOOTER ================
val footerStyle = MaterialTheme.typography.bodySmall.copy(
    color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.75f)
)

Column(
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 12.dp),
    horizontalAlignment = Alignment.CenterHorizontally
) {
    Text("Copyright ©2025", style = footerStyle)
    Text(
        "Praktikum #8 Menggunakan Multimedia",
        style = footerStyle.copy(fontWeight = FontWeight.Bold)
    )
    Text("Kuliah Mobile Programming S1 Teknologi Informasi", style = footerStyle)
    Text("UIN Antasari Banjarmasin", style = footerStyle)
```

```
            }
        }
        Spacer(Modifier.height(20.dp))
    }
}

// ================ COMPOSABLE HERO ================
@Composable
fun HeroCard(modifier: Modifier = Modifier) {
    Card(
        modifier = modifier.fillMaxWidth(),
        shape = RoundedCornerShape(18.dp),
        elevation = CardDefaults.cardElevation(defaultElevation = 8.dp)
    ) {
        Image(
            painter = painterResource(id = R.drawable.hero_multimedia),
            contentDescription = "Multimedia Hero",
            contentScale = ContentScale.Crop
        )
    }
}

// ================ COMPOSABLE MENU CARD ================
@Composable
fun MenuCard(
    icon: androidx.compose.ui.graphics.vector.ImageVector,
    title: String,
    modifier: Modifier = Modifier,
    onClick: () -> Unit
) {
    Card(
        modifier = modifier.height(120.dp),
        shape = RoundedCornerShape(16.dp),
        elevation = CardDefaults.cardElevation(6.dp),
        onClick = onClick
    ) {
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(12.dp),
            verticalArrangement = Arrangement.Center,
            horizontalAlignment = Alignment.CenterHorizontally
        ) {

            Box(
                modifier = Modifier
                    .size(54.dp)
                    .clip(CircleShape)
                    .background(Color(0xFF00C06B)),
                contentAlignment = Alignment.Center
            ) {
                Icon(icon, title, tint = Color.White, modifier = Modifier.size(28.dp))
            }
            Spacer(Modifier.height(10.dp))
            Text(title, style = MaterialTheme.typography.bodyMedium)
        }
    }
}
```

3.2 Siapkan file logo "logo_multimedia.png" letakkan di folder :
app/src/main/res/drawable/**logo_multimedia.jpg**

3.3 Siapkan file hero card "hero_multimedia.jpg" letakkan di folder :
app/src/main/res/drawable/**hero_multimedia.jpg**

3.4 Jalankan **Build** lalu **Run app** di emulator / device.

✓ Checklist
- ☐ UI Home tampil dengan (header gradient, logo bulat, hero card, grid menu).
- ☐ Semua card menu dapat ditekan dan navigasi berfungsi.
- ☐ Footer berada di bagian bawah konten.
- ☐ Saat HP di-rotate ke landscape, UI tetap rapi dan bisa di-scroll vertikal.
- ☐ Tidak ada overflow UI di landscape.
- ☐ Tidak ada error compiler / composable.

Langkah 4 — **Implementasi Audio Recorder & Audio Player** (MediaRecorder + ExoPlayer + Manajemen File) ----------

Target Belajar

A. **Audio Recorder** (MediaRecorder)
1. Menggunakan MediaRecorder untuk merekam audio dari microphone.
2. Menyimpan hasil rekaman dalam format .mp4 ke internal storage (context.filesDir).
3. Menangani izin runtime RECORD_AUDIO.
4. Mengimplementasikan tombol Start / Stop Recording.
5. Mengirim path file rekaman ke Audio Player melalui Navigation (encoded).

B. **Audio Player** (ExoPlayer)
1. Menggunakan ExoPlayer untuk memutar audio dari file lokal (File).
2. Membuat UI player lengkap:
3. Tombol Play / Pause
4. Slider durasi (seekTo)
5. Indikator waktu berjalan & total durasi
6. Mengganti file yang diputar secara otomatis ketika user memilih file lain.

C. **Manajemen File** Rekaman
1. Menampilkan daftar rekaman audio dalam bentuk Card rapi
2. Menampilkan informasi: filename, durasi, ukuran file
3. Fitur Edit nama file
4. Fitur Delete file
5. Menampilkan daftar rekaman baik di Recorder maupun Player
6. Menjadikan tampilan scrollable baik portrait maupun landscape

Langkah Praktikum

4.1 Update file : app/**build.gradle.kts** ---> Tambahkan dependency **ExoPlayer**

```
implementation("androidx.media3:media3-exoplayer:1.3.0")
implementation("androidx.media3:media3-ui:1.3.0")
```

✓ Di Android Studio: Klik **File → Sync Project with Gradle Files.**

4.2 Update file : util/**FileManagerUtility.kt**

```kotlin
package id.antasari.p8_multimedia_nimanda.util

import android.content.Context
import android.net.Uri
import androidx.media3.common.MediaItem
import androidx.media3.exoplayer.ExoPlayer
import java.io.File
import kotlin.math.roundToInt

data class AudioFileData(
    val file: File,
    val durationMs: Long,
    val sizeText: String
)

object FileManagerUtility {

    /** Ambil seluruh file rekaman dari internal storage */
    fun getAllAudioFiles(context: Context): List<File> {
        return context.filesDir.listFiles()
            ?.filter { it.extension.lowercase() == "mp4" }
            ?.sortedByDescending { it.lastModified() }
            ?: emptyList()
    }

    /** Convert bytes → KB/MB */
    fun formatFileSize(bytes: Long): String {
        val kb = bytes / 1024f
        return if (kb > 1024)
            "${(kb / 1024).roundToInt()} MB"
        else
            "${kb.roundToInt()} KB"
    }

    /** Mengambil durasi file menggunakan ExoPlayer tanpa memutar audio */
    fun getAudioDuration(context: Context, file: File): Long {
        val player = ExoPlayer.Builder(context).build()
        return try {
            player.setMediaItem(MediaItem.fromUri(Uri.fromFile(file)))
            player.prepare()
            val duration = player.duration
            player.release()
            duration
        } catch (e: Exception) {
            player.release()
            0L
        }
    }

    /** Rename file */
    fun renameFile(oldFile: File, newName: String): Boolean {
        val newFile = File(oldFile.parent, newName)
        return oldFile.renameTo(newFile)
    }

    /** Delete file */
    fun deleteFile(file: File): Boolean {
        return file.delete()
    }
}
```

4.3 Update file : ui/recorder/**AudioRecorderScreen.kt**

```
package id.antasari.p8_multimedia_nimanda.ui.recorder

import android.Manifest
import android.app.Activity
import android.content.pm.PackageManager
import android.media.MediaRecorder
import android.net.Uri
import android.os.Build
import android.util.Log
import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.core.app.ActivityCompat
import id.antasari.p8_multimedia_nimanda.ui.Screen
import id.antasari.p8_multimedia_nimanda.util.AudioFileData
import id.antasari.p8_multimedia_nimanda.util.FileManagerUtility
import java.io.File

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun AudioRecorderScreen(
    onBack: () -> Unit,
    onNavigate: (String) -> Unit
) {

    val context = LocalContext.current
    val activity = context as Activity
    val scrollState = rememberScrollState()

    // Permission
    LaunchedEffect(Unit) {
        if (ActivityCompat.checkSelfPermission(
                context, Manifest.permission.RECORD_AUDIO
            ) != PackageManager.PERMISSION_GRANTED
        ) {
            ActivityCompat.requestPermissions(
                activity,
                arrayOf(Manifest.permission.RECORD_AUDIO),
                101
            )
        }
    }
```

```
var isRecording by remember { mutableStateOf(false) }
var recorder: MediaRecorder? by remember { mutableStateOf(null) }
var outputFile by remember { mutableStateOf("") }

// List file
var audioFiles by remember { mutableStateOf(loadAudioFiles(context)) }

// Rename dialog
var showRenameDialog by remember { mutableStateOf<File?>(null) }
var newFileName by remember { mutableStateOf("") }

// Delete dialog
var showDeleteDialog by remember { mutableStateOf<File?>(null) }

// =============== START RECORDING ===============
fun startRecording() {
    val fileName = "audio_${System.currentTimeMillis()}.mp4"
    val file = File(context.filesDir, fileName)
    outputFile = file.absolutePath

    recorder = if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.S)
        MediaRecorder(context)
    else MediaRecorder()

    try {
        recorder?.apply {
            setAudioSource(MediaRecorder.AudioSource.MIC)
            setOutputFormat(MediaRecorder.OutputFormat.MPEG_4)
            setAudioEncoder(MediaRecorder.AudioEncoder.AAC)
            setOutputFile(outputFile)
            prepare()
            start()
            isRecording = true
        }
    } catch (e: Exception) {
        Log.e("Recorder", "Start error: ${e.message}")
        isRecording = false
    }
}

// =============== STOP RECORDING ===============
fun stopRecording() {
    try {
        if (isRecording) recorder?.stop()
    } catch (e: Exception) {
        Log.e("Recorder", "Stop error: ${e.message}")
    }

    recorder?.release()
    recorder = null
    isRecording = false

    val size = File(outputFile).length()

    if (size > 1500) {
        val encoded = Uri.encode(outputFile)
        onNavigate(Screen.AudioPlayer.passPath(encoded))
    }
```

```
    audioFiles = loadAudioFiles(context)
}

// ======================== UI ========================

Column(modifier = Modifier.fillMaxSize()) {

    TopAppBar(
        title = { Text("Audio Recorder") },
        navigationIcon = {
            IconButton(onClick = onBack) {
                Icon(Icons.Default.ArrowBack, contentDescription = null)
            }
        }
    )

    Column(
        modifier = Modifier
            .fillMaxWidth()
            .verticalScroll(scrollState)   // <--- FIX LANDSCAPE + PORTRAIT SCROLL
            .padding(horizontal = 20.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {

        Spacer(Modifier.height(20.dp))

        Text(
            if (isRecording) "Recording..." else "Ready to Record",
            color = if (isRecording) Color.Red else MaterialTheme.colorScheme.onSurface,
            style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold),
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth()
        )

        Spacer(Modifier.height(20.dp))

        FloatingActionButton(
            onClick = {
                if (!isRecording) startRecording()
                else stopRecording()
            },
            shape = CircleShape,
            modifier = Modifier.size(100.dp),
            containerColor =
                if (isRecording) Color.Red else MaterialTheme.colorScheme.primary
        ) {
            Icon(Icons.Default.Mic, contentDescription = null, tint = Color.White, modifier = Modifier.size(40.dp))
        }

        Spacer(Modifier.height(20.dp))

        Button(
            onClick = {
                if (!isRecording) startRecording()
                else stopRecording()
            },
            modifier = Modifier
                .fillMaxWidth()
                .height(48.dp)
```

```
        ) {
            Text(if (isRecording) "Stop Recording" else "Start Recording")
        }

        Spacer(Modifier.height(24.dp))

        Divider()
        Spacer(Modifier.height(16.dp))

        Text(
            "Daftar Rekaman",
            style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold)
        )

        Spacer(Modifier.height(12.dp))

        // ==== LIST FILE (MANUAL COLUMN) ====
        audioFiles.forEach { item ->

            FileCard(
                data = item,
                onPlay = {
                    val encoded = Uri.encode(item.file.absolutePath)
                    onNavigate(Screen.AudioPlayer.passPath(encoded))
                },
                onEdit = {
                    showRenameDialog = item.file
                    newFileName = item.file.nameWithoutExtension
                },
                onDelete = {
                    showDeleteDialog = item.file
                }
            )

             Spacer(Modifier.height(12.dp))
        }

        Spacer(Modifier.height(30.dp))
    }
}

// ==================== RENAME DIALOG ====================
if (showRenameDialog != null) {
    val file = showRenameDialog!!

    AlertDialog(
        onDismissRequest = { showRenameDialog = null },
        title = { Text("Edit Nama File") },
        text = {
            OutlinedTextField(
                value = newFileName,
                onValueChange = { newFileName = it },
                label = { Text("Nama baru (tanpa .mp4)") }
            )
        },
```

```
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.renameFile(file, "$newFileName.mp4")
                    audioFiles = loadAudioFiles(context)
                    showRenameDialog = null
                }) { Text("Simpan") }
            },
            dismissButton = {
                TextButton(onClick = { showRenameDialog = null }) { Text("Batal") }
            }
        )
    }

    // ==================== DELETE DIALOG ====================
    if (showDeleteDialog != null) {
        val file = showDeleteDialog!!

        AlertDialog(
            onDismissRequest = { showDeleteDialog = null },
            title = { Text("Hapus File?") },
            text = { Text(file.name) },
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.deleteFile(file)
                    audioFiles = loadAudioFiles(context)
                    showDeleteDialog = null
                }) { Text("Yes") }
            },
            dismissButton = {
                TextButton(onClick = { showDeleteDialog = null }) { Text("Cancel") }
            }
        )
    }
}

@Composable
fun FileCard(
    data: AudioFileData,
    onPlay: () -> Unit,
    onEdit: () -> Unit,
    onDelete: () -> Unit
) {

    Card(
        modifier = Modifier
            .fillMaxWidth()
            .clickable { onPlay() },
        colors = CardDefaults.cardColors(containerColor = Color(0xFFEFF7EE)),
        shape = RoundedCornerShape(12.dp),
        elevation = CardDefaults.cardElevation(3.dp)
    ) {
        Column(Modifier.padding(14.dp)) {

            Row(verticalAlignment = Alignment.CenterVertically) {
                Icon(
                    Icons.Default.AudioFile,
                    contentDescription = null,
                    tint = MaterialTheme.colorScheme.primary,
                    modifier = Modifier.size(34.dp)
                )
```

```
                Spacer(Modifier.width(12.dp))

                Text(
                    data.file.name,
                    fontWeight = FontWeight.Bold,
                    style = MaterialTheme.typography.bodyLarge
                )
            }

            Spacer(Modifier.height(6.dp))

            Text(
                "${formatDuration(data.durationMs)} • ${data.sizeText}",
                style = MaterialTheme.typography.bodySmall,
                color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.7f),
                modifier = Modifier.padding(start = 46.dp)
            )

            Spacer(Modifier.height(6.dp))

            Row(modifier = Modifier.padding(start = 46.dp)) {

                Text(
                    "[Edit]",
                    color = MaterialTheme.colorScheme.primary,
                    modifier = Modifier.clickable { onEdit() },
                    style = MaterialTheme.typography.bodySmall
                )

                Spacer(Modifier.width(20.dp))

                Text(
                    "[Delete]",
                    color = Color.Red,
                    modifier = Modifier.clickable { onDelete() },
                    style = MaterialTheme.typography.bodySmall
                )
            }
        }
    }
}

fun loadAudioFiles(context: android.content.Context): List<AudioFileData> {
    return FileManagerUtility.getAllAudioFiles(context).map { file ->
        AudioFileData(
            file,
            durationMs = FileManagerUtility.getAudioDuration(context, file),
            sizeText = FileManagerUtility.formatFileSize(file.length())
        )
    }
}

fun formatDuration(ms: Long): String {
    val sec = (ms / 1000)
    val min = sec / 60
    val s = sec % 60
    return "%02d:%02d".format(min, s)
}
```

4.4 Update file : ui/player/ **AudioPlayerScreen.kt**

```
package id.antasari.p8_multimedia_nimanda.ui.player

import android.net.Uri
import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.media3.common.MediaItem
import androidx.media3.exoplayer.ExoPlayer
import id.antasari.p8_multimedia_nimanda.ui.Screen
import id.antasari.p8_multimedia_nimanda.util.AudioFileData
import id.antasari.p8_multimedia_nimanda.util.FileManagerUtility
import kotlinx.coroutines.delay
import java.io.File

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun AudioPlayerScreen(
    onBack: () -> Unit,
    audioPath: String
) {
    val context = LocalContext.current
    val scrollState = rememberScrollState()

    var currentFile by remember { mutableStateOf(File(audioPath)) }
    var audioFiles by remember { mutableStateOf(loadAudioFiles(context)) }

    var showRenameDialog by remember { mutableStateOf<File?>(null) }
    var newFileName by remember { mutableStateOf("") }
    var showDeleteDialog by remember { mutableStateOf<File?>(null) }

    // Player
    val player = remember {
        ExoPlayer.Builder(context).build().apply {
            setMediaItem(MediaItem.fromUri(Uri.fromFile(currentFile)))
            prepare()
            play()
        }
    }

    var isPlaying by remember { mutableStateOf(true) }
    var position by remember { mutableStateOf(0L) }
    var duration by remember { mutableStateOf(1L) }
```

```
// Update slider
LaunchedEffect(isPlaying) {
    while (true) {
        if (player.isPlaying) {
            position = player.currentPosition
            duration = player.duration.coerceAtLeast(1)
        }
        delay(200)
    }
}

DisposableEffect(Unit) { onDispose { player.release() } }

LaunchedEffect(currentFile) {
    player.stop()
    player.setMediaItem(MediaItem.fromUri(Uri.fromFile(currentFile)))
    player.prepare()
    player.play()
    isPlaying = true
}

// ================ UI ====================

Column(modifier = Modifier.fillMaxSize()) {

    TopAppBar(
        title = { Text("Audio Player") },
        navigationIcon = {
            IconButton(onClick = onBack) {
                Icon(Icons.Default.ArrowBack, contentDescription = null)
            }
        }
    )
    Column(
        modifier = Modifier
            .fillMaxWidth()
            .verticalScroll(scrollState)   // <--- FIX SCROLL PORTRAIT/LANDSCAPE
            .padding(horizontal = 20.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {

        Spacer(Modifier.height(20.dp))

        Text(
            currentFile.name,
            style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold),
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth()
        )

        Spacer(Modifier.height(20.dp))

        FloatingActionButton(onClick = {
            if (isPlaying) {
                player.pause()
                isPlaying = false
            } else {
                player.play()
                isPlaying = true
            }
        }) {
```

```
            Icon(
                if (isPlaying) Icons.Default.Pause else Icons.Default.PlayArrow,
                contentDescription = null,
                modifier = Modifier.size(40.dp)
            )
        }

        Spacer(Modifier.height(20.dp))

        Slider(
            value = position.toFloat(),
            onValueChange = {
                position = it.toLong()
                player.seekTo(position)
            },
            valueRange = 0f..duration.toFloat(),
            modifier = Modifier.fillMaxWidth()
        )

        Row(
            Modifier.fillMaxWidth(),
            horizontalArrangement = Arrangement.SpaceBetween
        ) {
            Text(formatDuration(position))
            Text(formatDuration(duration))
        }

        Spacer(Modifier.height(20.dp))

        Divider()
        Spacer(Modifier.height(12.dp))

        Text(
            "Daftar Rekaman",
            style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold)
        )

        Spacer(Modifier.height(12.dp))

        // ==== LIST FILE ====
        audioFiles.forEach { item ->

            FileCard(
                data = item,
                onPlay = { currentFile = item.file },
                onEdit = {
                    showRenameDialog = item.file
                    newFileName = item.file.nameWithoutExtension
                },
                onDelete = {
                    showDeleteDialog = item.file
                }
            )

            Spacer(Modifier.height(12.dp))
        }

        Spacer(Modifier.height(30.dp))
    }
}
```

```
    // ------------ DIALOG RENAME ------------
    if (showRenameDialog != null) {
        val file = showRenameDialog!!

        AlertDialog(
            onDismissRequest = { showRenameDialog = null },
            title = { Text("Edit Nama File") },
            text = {
                OutlinedTextField(
                    value = newFileName,
                    onValueChange = { newFileName = it },
                    label = { Text("Nama baru (tanpa .mp4)") }
                )
            },
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.renameFile(file, "$newFileName.mp4")
                    audioFiles = loadAudioFiles(context)
                    if (file == currentFile) {
                        currentFile = File(context.filesDir, "$newFileName.mp4")
                    }
                    showRenameDialog = null
                }) { Text("Simpan") }
            },
            dismissButton = {
                TextButton(onClick = { showRenameDialog = null }) { Text("Batal") }
            }
        )
    }

    // ------------ DIALOG DELETE ------------
    if (showDeleteDialog != null) {
        val file = showDeleteDialog!!

        AlertDialog(
            onDismissRequest = { showDeleteDialog = null },
            title = { Text("Hapus File?") },
            text = { Text(file.name) },
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.deleteFile(file)
                    audioFiles = loadAudioFiles(context)
                    if (file == currentFile) {
                        if (audioFiles.isNotEmpty()) {
                            currentFile = audioFiles.first().file
                        } else {
                            player.pause()
                        }
                    }
                    showDeleteDialog = null
                }) { Text("Yes") }
            },
            dismissButton = {
                TextButton(onClick = { showDeleteDialog = null }) {
                    Text("Cancel")
                }
            }
        )
    }
}
```

```
@Composable
fun FileCard(
    data: AudioFileData,
    onPlay: () -> Unit,
    onEdit: () -> Unit,
    onDelete: () -> Unit
) {

    Card(
        modifier = Modifier
            .fillMaxWidth()
            .clickable { onPlay() },
        shape = RoundedCornerShape(12.dp),
        colors = CardDefaults.cardColors(containerColor = Color(0xFFEFF7EE)),
        elevation = CardDefaults.cardElevation(3.dp)
    ) {

        Column(Modifier.padding(14.dp)) {

            Row(verticalAlignment = Alignment.CenterVertically) {

                Icon(
                    Icons.Default.AudioFile,
                    contentDescription = null,
                    tint = MaterialTheme.colorScheme.primary,
                    modifier = Modifier.size(34.dp)
                )

                Spacer(Modifier.width(12.dp))

                Text(
                    data.file.name,
                    fontWeight = FontWeight.Bold,
                    style = MaterialTheme.typography.bodyLarge
                )
            }

            Spacer(Modifier.height(6.dp))

            Text(
                "${formatDuration(data.durationMs)} • ${data.sizeText}",
                style = MaterialTheme.typography.bodySmall,
                color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.7f),
                modifier = Modifier.padding(start = 46.dp)
            )

            Spacer(Modifier.height(6.dp))

            Row(modifier = Modifier.padding(start = 46.dp)) {

                Text(
                    "[Edit]",
                    modifier = Modifier.clickable { onEdit() },
                    color = MaterialTheme.colorScheme.primary,
                    style = MaterialTheme.typography.bodySmall
                )

                Spacer(Modifier.width(20.dp))
```

```
                Text(
                    "[Delete]",
                    modifier = Modifier.clickable { onDelete() },
                    color = Color.Red,
                    style = MaterialTheme.typography.bodySmall
                )
            }
        }
    }
}

fun loadAudioFiles(context: android.content.Context): List<AudioFileData> {
    return FileManagerUtility.getAllAudioFiles(context).map { file ->
        AudioFileData(
            file,
            durationMs = FileManagerUtility.getAudioDuration(context, file),
            sizeText = FileManagerUtility.formatFileSize(file.length())
        )
    }
}

fun formatDuration(ms: Long): String {
    val sec = (ms / 1000)
    val min = sec / 60
    val s = sec % 60
    return "%02d:%02d".format(min, s)
}
```

4.5 Update file : ui/ **NavGraph.kt**

```
package id.antasari.p8_multimedia_nimanda.ui

import android.net.Uri
import androidx.compose.runtime.Composable
import androidx.navigation.NavType
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import androidx.navigation.compose.rememberNavController
import androidx.navigation.navArgument
import id.antasari.p8_multimedia_nimanda.ui.gallery.CameraGalleryScreen
import id.antasari.p8_multimedia_nimanda.ui.home.HomeScreen
import id.antasari.p8_multimedia_nimanda.ui.player.AudioPlayerScreen
import id.antasari.p8_multimedia_nimanda.ui.recorder.AudioRecorderScreen
import id.antasari.p8_multimedia_nimanda.ui.video.VideoPlayerScreen

@Composable
fun AppNavHost(startDestination: String = Screen.Home.route) {

    val navController = rememberNavController()

    NavHost(
        navController = navController,
        startDestination = startDestination
    ) {
```

```
        // ====================== HOME ======================
        composable(Screen.Home.route) {
            HomeScreen(
                onNavigate = { route ->
                    navController.navigate(route)
                }
            )
        }

        // ====================== AUDIO RECORDER ======================
        composable(Screen.AudioRecorder.route) {
            AudioRecorderScreen(
                onBack = { navController.popBackStack() },
                onNavigate = { encodedRoute ->
                    navController.navigate(encodedRoute)
                }
            )
        }

        // ====================== AUDIO PLAYER (ENCODED ARGUMENT) ======================
        composable(
            route = Screen.AudioPlayer.route,
            arguments = listOf(
                navArgument("audioPath") {
                    type = NavType.StringType
                    nullable = false
                }
            )
        ) { backStackEntry ->

            val encodedPath = backStackEntry.arguments?.getString("audioPath") ?: ""

            // Decode kembali agar ExoPlayer bisa membaca file asli
            val audioPath = Uri.decode(encodedPath)

            AudioPlayerScreen(
                onBack = { navController.popBackStack() },
                audioPath = audioPath
            )
        }

        // ====================== VIDEO PLAYER ======================
        composable(Screen.VideoPlayer.route) {
            VideoPlayerScreen(
                onBack = { navController.popBackStack() }
            )
        }

        // ====================== CAMERA & GALLERY ======================
        composable(Screen.CameraGallery.route) {
            CameraGalleryScreen(
                onBack = { navController.popBackStack() }
            )
        }
    }
}
```

4.6 Update file : ui/**Screen.kt**

```
package id.antasari.p8_multimedia_nimanda.ui

sealed class Screen(val route: String) {

    object Home : Screen("home")
    object AudioRecorder : Screen("audio_recorder")

    object AudioPlayer : Screen("audio_player/{audioPath}") {
        fun passPath(encoded: String): String = "audio_player/$encoded"
    }

    object VideoPlayer : Screen("video_player")
    object CameraGallery : Screen("camera_gallery")
}
```

4.7 Jalankan **Build** lalu **Run app** di emulator / device.

✓ Checklist

A. Audio Recorder
- ☐ Aplikasi meminta izin RECORD_AUDIO saat pertama dibuka
- ☐ Tombol Start Recording berfungsi
- ☐ File rekaman tersimpan dalam format .mp4
- ☐ Selesai recording tidak menyebabkan crash
- ☐ File rekaman tampil dalam daftar rekaman
- ☐ Durasi rekaman tampil dengan benar
- ☐ Ukuran file rekaman tampil dengan benar
- ☐ Ketika file ditekan → pindah ke Audio Player dan memutar file tersebut

B. Navigasi Recorder → Player
- ☐ Path file dikirim menggunakan Uri.encode()
- ☐ AudioPlayer menerima argument audioPath dengan benar
- ☐ File otomatis diputar saat masuk ke AudioPlayer

C. Audio Player (ExoPlayer)
- ☐ Tombol Play/Pause bekerja
- ☐ Slider durasi dapat digeser
- ☐ Audio berpindah posisi sesuai slider
- ☐ Waktu berjalan dan durasi total tampil dengan benar
- ☐ Memilih file lain dari daftar → langsung memutar file tersebut
- ☐ Player tidak crash ketika rotasi layar (portrait ↔ landscape)

D. Manajemen File Rekaman
- ☐ Daftar rekaman tampil di Recorder dan Player
- ☐ Tekan [Edit] → dialog rename muncul
- ☐ Rename file berhasil dan update di daftar
- ☐ Tekan [Delete] → dialog konfirmasi muncul
- ☐ File benar-benar terhapus
- ☐ Jika file yang sedang diputar dihapus → Player berpindah ke file lain atau berhenti

E. UI & Layout (Portrait + Landscape)
- ☐ Semua tampilan bisa di-scroll saat portrait
- ☐ Semua tampilan bisa di-scroll saat landscape
- ☐ Layout tidak terpotong pada mode landscape
- ☐ Jarak antar elemen tetap rapi
- ☐ UI tetap konsisten antara Recorder dan Player

Langkah 5 — Implementasi Lanjutan Multimedia Interaktif (Camera, Gallery, Audio & Video) ----------

Target Belajar

1. Mengimplementasikan Camera & Gallery Screen dengan antarmuka berbasis Card Menu.
2. Menampilkan preview photo yang:
   - ✓ dapat di-zoom (pinch zoom),
   - ✓ dapat di-geser (pan ke atas, bawah, kiri, kanan),
   - ✓ tetap tampil saat rotasi layar (portrait ↔ landscape),
   - ✓ dapat disimpan permanen ke Galeri HP.
3. Menampilkan Audio Recorder & Audio Player:
   - ✓ hanya menampilkan file audio,
   - ✓ audio dapat diputar otomatis sesuai pilihan file.
4. Menampilkan Video Player dengan fitur:
   - ✓ pemutaran video berbasis ExoPlayer,
   - ✓ fullscreen mode,
   - ✓ zoom dan pan pada video preview untuk melihat detail video.
5. Mengintegrasikan gesture multimedia (pinch, pan, double tap) pada konten image dan video menggunakan Jetpack Compose.
6. Mengelola file multimedia (rename, delete, play) secara aman di internal storage.

Langkah Praktikum

5.1 Update file : ui/gallery/**CameraGalleryScreen**.kt

```
package id.antasari.p8_multimedia_nimanda.ui.gallery

import android.content.ContentValues
import android.content.Context
import android.content.Intent
import android.content.res.Configuration
import android.graphics.Bitmap
import android.graphics.BitmapFactory
import android.net.Uri
import android.os.Build
import android.provider.MediaStore
import android.widget.Toast
import androidx.activity.compose.rememberLauncherForActivityResult
import androidx.activity.result.contract.ActivityResultContracts
import androidx.compose.foundation.BorderStroke
import androidx.compose.foundation.Image
import androidx.compose.foundation.clickable
import androidx.compose.foundation.gestures.detectTransformGestures
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.graphics.asImageBitmap
import androidx.compose.ui.graphics.graphicsLayer
import androidx.compose.ui.input.pointer.pointerInput
import androidx.compose.ui.platform.LocalConfiguration
```

```
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import id.antasari.p8_multimedia_nimanda.ui.Screen
import java.io.File
import java.io.FileOutputStream
import java.io.OutputStream

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun CameraGalleryScreen(
    onBack: () -> Unit,
    onNavigate: (String) -> Unit
) {

    val context = LocalContext.current
    val configuration = LocalConfiguration.current
    val scrollState = rememberScrollState()

    var selectedBitmap by remember { mutableStateOf<Bitmap?>(null) }
    var showCameraOption by remember { mutableStateOf(false) }

    // ===== TRANSFORM STATE =====
    var scale by remember { mutableStateOf(1f) }
    var offsetX by remember { mutableStateOf(0f) }
    var offsetY by remember { mutableStateOf(0f) }

    val isLandscape =
        configuration.orientation == Configuration.ORIENTATION_LANDSCAPE

    // ================ CAMERA ================

    val takePhotoLauncher =
        rememberLauncherForActivityResult(ActivityResultContracts.TakePicturePreview()) { bitmap ->
            bitmap?.let {
                selectedBitmap = it
                scale = 1f
                offsetX = 0f
                offsetY = 0f
            }
        }

    val recordVideoLauncher =
        rememberLauncherForActivityResult(ActivityResultContracts.StartActivityForResult()) { result ->
            if (result.resultCode == android.app.Activity.RESULT_OK) {
                result.data?.data?.let { uri ->
                    val file = copyToInternal(context, uri, "video_", ".mp4")
                    onNavigate(Screen.VideoPlayer.passPath(Uri.encode(file.absolutePath)))
                }
            }
        }
```

```
// ================= GALLERY =================

val pickMediaLauncher =
    rememberLauncherForActivityResult(ActivityResultContracts.GetContent()) { uri ->
        uri?.let {
            val type = context.contentResolver.getType(it) ?: ""
            when {
                type.startsWith("image") -> {
                    selectedBitmap = loadBitmap(context, it)
                    scale = 1f
                    offsetX = 0f
                    offsetY = 0f
                }
                type.startsWith("video") -> {
                    val file = copyToInternal(context, it, "video_", ".mp4")
                    onNavigate(Screen.VideoPlayer.passPath(Uri.encode(file.absolutePath)))
                }
                type.startsWith("audio") -> {
                    val file = copyToInternal(context, it, "audio_", ".mp4")
                    onNavigate(Screen.AudioPlayer.passPath(Uri.encode(file.absolutePath)))
                }
            }
        }
    }
// ================= UI =================

Column(modifier = Modifier.fillMaxSize()) {

    TopAppBar(
        title = { Text("Camera & Gallery") },
        navigationIcon = {
            IconButton(onClick = onBack) {
                Icon(Icons.Default.ArrowBack, contentDescription = null)
            }
        }
    )

    Column(
        modifier = Modifier
            .fillMaxWidth()
            .verticalScroll(scrollState)
            .padding(12.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {

        Spacer(Modifier.height(12.dp))
```

```
// ===== CARD MENU : OPEN CAMERA =====
Card(
    modifier = Modifier
        .fillMaxWidth()
        .clickable { showCameraOption = true },
    shape = RoundedCornerShape(16.dp),
    colors = CardDefaults.cardColors(containerColor = Color(0xFF22C55E))
) {
    Row(
        modifier = Modifier.padding(16.dp),
        verticalAlignment = Alignment.CenterVertically
    ) {
        Icon(
            Icons.Default.CameraAlt,
            contentDescription = null,
            tint = Color.White,
            modifier = Modifier.size(48.dp)
        )
        Spacer(Modifier.width(16.dp))
        Column {
            Text("Open Camera", fontWeight = FontWeight.Bold, color = Color.White)
            Text(
                "Take a new photo or Video",
                style = MaterialTheme.typography.bodySmall,
                color = Color.White
            )
        }
    }
}
Spacer(Modifier.height(16.dp))

// ===== CARD MENU : CHOOSE FROM GALLERY =====
Card(
    modifier = Modifier
        .fillMaxWidth()
        .clickable { pickMediaLauncher.launch("*/*") },
    shape = RoundedCornerShape(16.dp),
    border = BorderStroke(1.dp, Color(0xFF22C55E))
) {
    Row(
        modifier = Modifier.padding(16.dp),
        verticalAlignment = Alignment.CenterVertically
    ) {
        Icon(
            Icons.Default.Image,
            contentDescription = null,
            tint = Color(0xFF22C55E),
            modifier = Modifier.size(48.dp)
        )
        Spacer(Modifier.width(16.dp))
        Column {
            Text("Choose from Gallery", fontWeight = FontWeight.Bold)
            Text(
                "Select existing photo or video",
                style = MaterialTheme.typography.bodySmall
            )
        }
    }
}

Spacer(Modifier.height(20.dp))
```

```
// ===== IMAGE PREVIEW (ZOOM + PAN) =====
if (selectedBitmap != null) {
    Card(
        modifier = Modifier.fillMaxWidth(),
        colors = CardDefaults.cardColors(containerColor = Color(0xFFEFFAF3)),
        shape = RoundedCornerShape(16.dp)
    ) {
        Image(
            bitmap = selectedBitmap!!.asImageBitmap(),
            contentDescription = null,
            modifier = Modifier
                .fillMaxWidth()
                .height(if (isLandscape) 420.dp else 360.dp)
                .graphicsLayer(
                    scaleX = scale,
                    scaleY = scale,
                    translationX = offsetX,
                    translationY = offsetY
                )
                .pointerInput(Unit) {
                    detectTransformGestures { _, pan, zoom, _ ->
                        scale = (scale * zoom).coerceIn(1f, 5f)
                        offsetX += pan.x
                        offsetY += pan.y
                    }
                }
                .padding(12.dp)
        )
    }

    Spacer(Modifier.height(12.dp))

    Button(
        onClick = {
            selectedBitmap?.let { saveImageToGallery(context, it) }
        },
        modifier = Modifier.fillMaxWidth()
    ) {
        Icon(Icons.Default.Save, contentDescription = null)
        Spacer(Modifier.width(8.dp))
        Text("Simpan ke Galeri")
    }
}
// ===== EMPTY STATE =====
if (selectedBitmap == null) {
    Card(
        modifier = Modifier
            .fillMaxWidth()
            .height(360.dp),
        colors = CardDefaults.cardColors(containerColor = Color(0xFFEFFAF3)),
        shape = RoundedCornerShape(16.dp)
    ) {
```

```
                    Column(
                        modifier = Modifier.fillMaxSize(),
                        horizontalAlignment = Alignment.CenterHorizontally,
                        verticalArrangement = Arrangement.Center
                    ) {
                        Icon(
                            Icons.Default.Image,
                            contentDescription = null,
                            tint = Color(0xFF22C55E),
                            modifier = Modifier.size(48.dp)
                        )
                        Spacer(Modifier.height(8.dp))
                        Text("No image selected", style = MaterialTheme.typography.titleMedium)
                        Text(
                            "Choose an option above to get started",
                            style = MaterialTheme.typography.bodySmall,
                            textAlign = TextAlign.Center
                        )
                    }
                }
            }

            Spacer(Modifier.height(30.dp))

            Text(
                "Copyright ©2025\n" +
                        "Praktikum #8 Menggunakan Multimedia\n" +
                        "S1 Teknologi Informasi UIN Antasari\n" +
                        "Banjarmasin",
                modifier = Modifier.fillMaxWidth(),
                textAlign = TextAlign.Center,
                style = MaterialTheme.typography.bodySmall
            )
        }
    }
    // ===== CAMERA OPTION =====
    if (showCameraOption) {
        AlertDialog(
            onDismissRequest = { showCameraOption = false },
            title = { Text("Open Camera") },
            text = { Text("Choose what you want to capture") },
            confirmButton = {
                TextButton(onClick = {
                    showCameraOption = false
                    takePhotoLauncher.launch(null)
                }) { Text("Photo") }
            },
            dismissButton = {
                TextButton(onClick = {
                    showCameraOption = false
                    recordVideoLauncher.launch(Intent(MediaStore.ACTION_VIDEO_CAPTURE))
                }) { Text("Video") }
            }
        )
    }
}
```

```
// ================= SAVE TO GALLERY =================

fun saveImageToGallery(context: Context, bitmap: Bitmap) {

    val filename = "IMG_${System.currentTimeMillis()}.jpg"
    val fos: OutputStream?

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.Q) {

        val contentValues = ContentValues().apply {
            put(MediaStore.Images.Media.DISPLAY_NAME, filename)
            put(MediaStore.Images.Media.MIME_TYPE, "image/jpeg")
            put(MediaStore.Images.Media.RELATIVE_PATH, "Pictures/CameraGallery")
        }

        val imageUri = context.contentResolver.insert(
            MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
            contentValues
        )

        fos = imageUri?.let {
            context.contentResolver.openOutputStream(it)
        }

    } else {
        fos = context.contentResolver.openOutputStream(
            MediaStore.Images.Media.EXTERNAL_CONTENT_URI
        )
    }

    fos?.use {
        bitmap.compress(Bitmap.CompressFormat.JPEG, 95, it)
    }

    Toast.makeText(context, "Foto berhasil disimpan ke Galeri", Toast.LENGTH_SHORT).show()
}

// ================= HELPERS =================

fun copyToInternal(context: Context, uri: Uri, prefix: String, ext: String): File {
    val input = context.contentResolver.openInputStream(uri)!!
    val file = File(context.filesDir, "$prefix${System.currentTimeMillis()}$ext")
    FileOutputStream(file).use { output -> input.copyTo(output) }
    input.close()
    return file
}

fun loadBitmap(context: Context, uri: Uri): Bitmap? {
    return context.contentResolver.openInputStream(uri)?.use {
        BitmapFactory.decodeStream(it)
    }
}
```

5.2 Update file : ui/recorder/**AudioRecorderScreen**.kt

```
package id.antasari.p8_multimedia_nimanda.ui.recorder

import android.Manifest
import android.app.Activity
import android.content.pm.PackageManager
import android.media.MediaRecorder
import android.net.Uri
import android.os.Build
import android.util.Log
import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.core.app.ActivityCompat
import id.antasari.p8_multimedia_nimanda.ui.Screen
import id.antasari.p8_multimedia_nimanda.util.AudioFileData
import id.antasari.p8_multimedia_nimanda.util.FileManagerUtility
import java.io.File
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun AudioRecorderScreen(
    onBack: () -> Unit,
    onNavigate: (String) -> Unit
) {

    val context = LocalContext.current
    val activity = context as Activity
    val scrollState = rememberScrollState()

    // Permission
    LaunchedEffect(Unit) {
        if (ActivityCompat.checkSelfPermission(
                context, Manifest.permission.RECORD_AUDIO
            ) != PackageManager.PERMISSION_GRANTED
        ) {
            ActivityCompat.requestPermissions(
                activity,
                arrayOf(Manifest.permission.RECORD_AUDIO),
                101
            )
        }
    }
```

```
var isRecording by remember { mutableStateOf(false) }
var recorder: MediaRecorder? by remember { mutableStateOf(null) }
var outputFile by remember { mutableStateOf("") }

// List file
var audioFiles by remember { mutableStateOf(loadAudioFiles(context)) }

// Rename dialog
var showRenameDialog by remember { mutableStateOf<File?>(null) }
var newFileName by remember { mutableStateOf("") }

// Delete dialog
var showDeleteDialog by remember { mutableStateOf<File?>(null) }

// =============== START RECORDING ===============
fun startRecording() {
    val fileName = "audio_${System.currentTimeMillis()}.mp4"
    val file = File(context.filesDir, fileName)
    outputFile = file.absolutePath

    recorder = if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.S)
        MediaRecorder(context)
    else MediaRecorder()

    try {
        recorder?.apply {
            setAudioSource(MediaRecorder.AudioSource.MIC)
            setOutputFormat(MediaRecorder.OutputFormat.MPEG_4)
            setAudioEncoder(MediaRecorder.AudioEncoder.AAC)
            setOutputFile(outputFile)
            prepare()
            start()
            isRecording = true
        }
    } catch (e: Exception) {
        Log.e("Recorder", "Start error: ${e.message}")
        isRecording = false
    }
}

// =============== STOP RECORDING ===============
fun stopRecording() {
    try {
        if (isRecording) recorder?.stop()
    } catch (e: Exception) {
        Log.e("Recorder", "Stop error: ${e.message}")
    }

    recorder?.release()
    recorder = null
    isRecording = false

    val size = File(outputFile).length()

    if (size > 1500) {
        val encoded = Uri.encode(outputFile)
        onNavigate(Screen.AudioPlayer.passPath(encoded))
    }

    audioFiles = loadAudioFiles(context)
}
```

```
// ======================== UI ========================

Column(modifier = Modifier.fillMaxSize()) {

    TopAppBar(
        title = { Text("Audio Recorder") },
        navigationIcon = {
            IconButton(onClick = onBack) {
                Icon(Icons.Default.ArrowBack, contentDescription = null)
            }
        }
    )

    Column(
        modifier = Modifier
            .fillMaxWidth()
            .verticalScroll(scrollState)   // <--- FIX LANDSCAPE + PORTRAIT SCROLL
            .padding(horizontal = 20.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {

        Spacer(Modifier.height(20.dp))

        Text(
            if (isRecording) "Recording..." else "Ready to Record",
            color = if (isRecording) Color.Red else MaterialTheme.colorScheme.onSurface,
            style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold),
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth()
        )

        Spacer(Modifier.height(20.dp))
      FloatingActionButton(
          onClick = {
              if (!isRecording) startRecording()
              else stopRecording()
          },
          shape = CircleShape,
          modifier = Modifier.size(100.dp),
          containerColor =
              if (isRecording) Color.Red else MaterialTheme.colorScheme.primary
      ) {
          Icon(Icons.Default.Mic, contentDescription = null, tint = Color.White, modifier = Modifier.size(40.dp))
      }

      Spacer(Modifier.height(20.dp))

      Button(
          onClick = {
              if (!isRecording) startRecording()
              else stopRecording()
          },
          modifier = Modifier
              .fillMaxWidth()
              .height(48.dp)
      ) {
          Text(if (isRecording) "Stop Recording" else "Start Recording")
      }
```

```
        Spacer(Modifier.height(24.dp))

        Divider()
        Spacer(Modifier.height(16.dp))

        Text(
            "Daftar Rekaman",
            style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold)
        )

        Spacer(Modifier.height(12.dp))

        // ==== LIST FILE (MANUAL COLUMN) ====
        audioFiles.forEach { item ->

            FileCard(
                data = item,
                onPlay = {
                    val encoded = Uri.encode(item.file.absolutePath)
                    onNavigate(Screen.AudioPlayer.passPath(encoded))
                },
                onEdit = {
                    showRenameDialog = item.file
                    newFileName = item.file.nameWithoutExtension
                },
                onDelete = {
                    showDeleteDialog = item.file
                }
            )

            Spacer(Modifier.height(12.dp))
        }
        Spacer(Modifier.height(30.dp))
    }
}

// ==================== RENAME DIALOG ====================
if (showRenameDialog != null) {
    val file = showRenameDialog!!

    AlertDialog(
        onDismissRequest = { showRenameDialog = null },
        title = { Text("Edit Nama File") },
        text = {
            OutlinedTextField(
                value = newFileName,
                onValueChange = { newFileName = it },
                label = { Text("Nama baru (tanpa .mp4)") }
            )
        },
        confirmButton = {
            TextButton(onClick = {
                FileManagerUtility.renameFile(file, "$newFileName.mp4")
                audioFiles = loadAudioFiles(context)
                showRenameDialog = null
            }) { Text("Simpan") }
        },
        dismissButton = {
            TextButton(onClick = { showRenameDialog = null }) { Text("Batal") }
        }
    )
}
```

```
    // ==================== DELETE DIALOG ====================
    if (showDeleteDialog != null) {
        val file = showDeleteDialog!!

        AlertDialog(
            onDismissRequest = { showDeleteDialog = null },
            title = { Text("Hapus File?") },
            text = { Text(file.name) },
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.deleteFile(file)
                    audioFiles = loadAudioFiles(context)
                    showDeleteDialog = null
                }) { Text("Yes") }
            },
            dismissButton = {
                TextButton(onClick = { showDeleteDialog = null }) { Text("Cancel") }
            }
        )
    }
}

@Composable
fun FileCard(
    data: AudioFileData,
    onPlay: () -> Unit,
    onEdit: () -> Unit,
    onDelete: () -> Unit
) {
    Card(
        modifier = Modifier
            .fillMaxWidth()
            .clickable { onPlay() },
        colors = CardDefaults.cardColors(containerColor = Color(0xFFEFF7EE)),
        shape = RoundedCornerShape(12.dp),
        elevation = CardDefaults.cardElevation(3.dp)
    ) {
        Column(Modifier.padding(14.dp)) {

            Row(verticalAlignment = Alignment.CenterVertically) {
                Icon(
                    Icons.Default.AudioFile,
                    contentDescription = null,
                    tint = MaterialTheme.colorScheme.primary,
                    modifier = Modifier.size(34.dp)
                )

                Spacer(Modifier.width(12.dp))

                Text(
                    data.file.name,
                    fontWeight = FontWeight.Bold,
                    style = MaterialTheme.typography.bodyLarge
                )
            }

            Spacer(Modifier.height(6.dp))
```

```
            Text(
                "${formatDuration(data.durationMs)} • ${data.sizeText}",
                style = MaterialTheme.typography.bodySmall,
                color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.7f),
                modifier = Modifier.padding(start = 46.dp)
            )

            Spacer(Modifier.height(6.dp))

            Row(modifier = Modifier.padding(start = 46.dp)) {

                Text(
                    "[Edit]",
                    color = MaterialTheme.colorScheme.primary,
                    modifier = Modifier.clickable { onEdit() },
                    style = MaterialTheme.typography.bodySmall
                )

                Spacer(Modifier.width(20.dp))

                Text(
                    "[Delete]",
                    color = Color.Red,
                    modifier = Modifier.clickable { onDelete() },
                    style = MaterialTheme.typography.bodySmall
                )
            }
        }
    }
}

fun loadAudioFiles(context: android.content.Context): List<AudioFileData> {

    val audioExtensions = listOf("mp3", "wav", "m4a", "mp4")

    return FileManagerUtility.getAllAudioFiles(context)
        .filter { file ->
            file.extension.lowercase() in audioExtensions
        }
        .map { file ->
            AudioFileData(
                file,
                durationMs = FileManagerUtility.getAudioDuration(context, file),
                sizeText = FileManagerUtility.formatFileSize(file.length())
            )
        }
}

fun formatDuration(ms: Long): String {
    val sec = (ms / 1000)
    val min = sec / 60
    val s = sec % 60
    return "%02d:%02d".format(min, s)
}
```

5.3 Update file : ui/video/**VideoPlayerScreen**.kt

```
package id.antasari.p8_multimedia_nimanda.ui.video

import android.app.Activity
import android.content.pm.ActivityInfo
import android.net.Uri
import androidx.compose.foundation.clickable
import androidx.compose.foundation.gestures.detectTransformGestures
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.graphics.graphicsLayer
import androidx.compose.ui.input.pointer.pointerInput
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.platform.LocalLifecycleOwner
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.viewinterop.AndroidView
import androidx.lifecycle.Lifecycle
import androidx.lifecycle.LifecycleEventObserver
import androidx.media3.common.MediaItem
import androidx.media3.exoplayer.ExoPlayer
import androidx.media3.ui.PlayerView
import id.antasari.p8_multimedia_nimanda.util.FileManagerUtility
import kotlinx.coroutines.delay
import java.io.File
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun VideoPlayerScreen(
    onBack: () -> Unit,
    videoPath: String
) {

    val context = LocalContext.current
    val activity = context as Activity
    val scrollState = rememberScrollState()

    // ================ STATE ================
    var currentFile by remember { mutableStateOf(File(videoPath)) }
    var videoFiles by remember { mutableStateOf(loadVideoFiles(context)) }

    var isFullscreen by remember { mutableStateOf(false) }
    var isPlaying by remember { mutableStateOf(true) }
    var position by remember { mutableStateOf(0L) }
    var duration by remember { mutableStateOf(1L) }
```

```
// ===== ZOOM & PAN STATE =====
var scale by remember { mutableStateOf(1f) }
var offsetX by remember { mutableStateOf(0f) }
var offsetY by remember { mutableStateOf(0f) }

// Dialog state
var showRenameDialog by remember { mutableStateOf<File?>(null) }
var newFileName by remember { mutableStateOf("") }
var showDeleteDialog by remember { mutableStateOf<File?>(null) }

// ================ PLAYER ================
val player = remember {
    ExoPlayer.Builder(context).build().apply {
        setMediaItem(MediaItem.fromUri(Uri.fromFile(currentFile)))
        prepare()
        play()
    }
}

// Update player when file changes
LaunchedEffect(currentFile) {
    player.stop()
    player.setMediaItem(MediaItem.fromUri(Uri.fromFile(currentFile)))
    player.prepare()
    player.play()
    isPlaying = true

    // reset transform
    scale = 1f
    offsetX = 0f
    offsetY = 0f
}
// Update slider
LaunchedEffect(isPlaying) {
    while (true) {
        if (player.isPlaying) {
            position = player.currentPosition
            duration = player.duration.coerceAtLeast(1L)
        }
        delay(200)
    }
}

// Lifecycle
val lifecycleOwner = LocalLifecycleOwner.current
DisposableEffect(Unit) {
    val observer = LifecycleEventObserver { _, event ->
        if (event == Lifecycle.Event.ON_PAUSE) {
            player.pause()
            isPlaying = false
        }
    }

    lifecycleOwner.lifecycle.addObserver(observer)
```

```
    onDispose {
        activity.requestedOrientation =
            ActivityInfo.SCREEN_ORIENTATION_UNSPECIFIED
        lifecycleOwner.lifecycle.removeObserver(observer)
        player.release()
    }
}

// ================ FULLSCREEN ================
fun enterFullscreen() {
    isFullscreen = true
    activity.requestedOrientation =
        ActivityInfo.SCREEN_ORIENTATION_SENSOR_LANDSCAPE
}

fun exitFullscreen() {
    isFullscreen = false
    activity.requestedOrientation =
        ActivityInfo.SCREEN_ORIENTATION_UNSPECIFIED
}

// ================ UI ================
Column(modifier = Modifier.fillMaxSize()) {

    if (!isFullscreen) {
        TopAppBar(
            title = { Text("Video Player") },
            navigationIcon = {
                IconButton(onClick = onBack) {
                    Icon(Icons.Default.ArrowBack, contentDescription = null)
                }
            },
             actions = {
                 IconButton(onClick = { enterFullscreen() }) {
                     Icon(Icons.Default.Fullscreen, contentDescription = null)
                 }
             }
         )
    }

    Column(
        modifier = Modifier
            .fillMaxWidth()
            .verticalScroll(scrollState)
    ) {

        // ================ VIDEO VIEW (ZOOM + PAN) ================
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(if (isFullscreen) 300.dp else 220.dp)
                .graphicsLayer(
                    scaleX = scale,
                    scaleY = scale,
                    translationX = offsetX,
                    translationY = offsetY
                )
```

```
            .pointerInput(Unit) {
                detectTransformGestures { _, pan, zoom, _ ->
                    scale = (scale * zoom).coerceIn(1f, 5f)
                    offsetX += pan.x
                    offsetY += pan.y
                }
            }
    ) {
        AndroidView(
            factory = { ctx ->
                PlayerView(ctx).also { view ->
                    view.player = player
                    view.useController = false
                }
            },
            modifier = Modifier.fillMaxSize()
        )
    }

    if (isFullscreen) {
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .padding(12.dp),
            horizontalArrangement = Arrangement.End
        ) {
            IconButton(onClick = { exitFullscreen() }) {
                Icon(Icons.Default.FullscreenExit, contentDescription = null)
            }
        }
    }

    Spacer(Modifier.height(20.dp))
    // ================ TITLE ================
    Text(
        currentFile.name,
        style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold),
        textAlign = TextAlign.Center,
        modifier = Modifier.fillMaxWidth()
    )

    Spacer(Modifier.height(20.dp))

    // ================ PLAY / PAUSE ================
    FloatingActionButton(
        onClick = {
            if (isPlaying) {
                player.pause()
                isPlaying = false
            } else {
                player.play()
                isPlaying = true
            }
        },
```

```
        modifier = Modifier.align(Alignment.CenterHorizontally)
    ) {
        Icon(
            if (isPlaying) Icons.Default.Pause else Icons.Default.PlayArrow,
            contentDescription = null,
            modifier = Modifier.size(40.dp)
        )
    }

    Spacer(Modifier.height(20.dp))

    // ================ SLIDER ================
    Slider(
        value = position.toFloat(),
        onValueChange = {
            position = it.toLong()
            player.seekTo(position)
        },
        valueRange = 0f..duration.toFloat(),
        modifier = Modifier.padding(horizontal = 20.dp)
    )

    Row(
        modifier = Modifier
            .fillMaxWidth()
            .padding(horizontal = 20.dp),
        horizontalArrangement = Arrangement.SpaceBetween
    ) {
        Text(formatDuration(position))
        Text(formatDuration(duration))
    }
    Spacer(Modifier.height(24.dp))
    Divider()
    Spacer(Modifier.height(14.dp))

    // ================ LIST VIDEO ================
    Text(
        "Daftar Video",
        style = MaterialTheme.typography.titleMedium.copy(fontWeight = FontWeight.Bold),
        modifier = Modifier.padding(horizontal = 20.dp)
    )

    Spacer(Modifier.height(12.dp))

    videoFiles.forEach { file ->
        VideoFileCard(
            file = file,
            onPlay = { currentFile = file },
            onEdit = {
                showRenameDialog = file
                newFileName = file.nameWithoutExtension
            },
```

```
                    onDelete = {
                        showDeleteDialog = file
                    }
                )
                Spacer(Modifier.height(12.dp))
            }

            Spacer(Modifier.height(40.dp))
        }
    }

    // ================ RENAME DIALOG ================
    if (showRenameDialog != null) {
        val file = showRenameDialog!!
        AlertDialog(
            onDismissRequest = { showRenameDialog = null },
            title = { Text("Edit Nama Video") },
            text = {
                OutlinedTextField(
                    value = newFileName,
                    onValueChange = { newFileName = it },
                    label = { Text("Nama baru (tanpa .mp4)") }
                )
            },
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.renameFile(file, "$newFileName.mp4")
                    videoFiles = loadVideoFiles(context)
                    currentFile = File(context.filesDir, "$newFileName.mp4")
                    showRenameDialog = null
                }) { Text("Simpan") }
            },
            dismissButton = {
                TextButton(onClick = { showRenameDialog = null }) { Text("Batal") }
            }
        )
    }

    // ================ DELETE DIALOG ================
    if (showDeleteDialog != null) {
        val file = showDeleteDialog!!
        AlertDialog(
            onDismissRequest = { showDeleteDialog = null },
            title = { Text("Hapus Video?") },
            text = { Text(file.name) },
            confirmButton = {
                TextButton(onClick = {
                    FileManagerUtility.deleteFile(file)
                    videoFiles = loadVideoFiles(context)
                    if (file == currentFile && videoFiles.isNotEmpty()) {
                        currentFile = videoFiles.first()
                    }
                    showDeleteDialog = null
                }) { Text("Yes") }
            },
            dismissButton = {
                TextButton(onClick = { showDeleteDialog = null }) { Text("Cancel") }
            }
        )
    }
}
```

```
// ================ HELPER ================
fun loadVideoFiles(context: android.content.Context): List<File> {
    return FileManagerUtility.getAllVideoFiles(context)
}

fun formatDuration(ms: Long): String {
    val sec = (ms / 1000)
    val min = sec / 60
    val s = sec % 60
    return "%02d:%02d".format(min, s)
}

// ================ CARD ================
@Composable
fun VideoFileCard(
    file: File,
    onPlay: () -> Unit,
    onEdit: () -> Unit,
    onDelete: () -> Unit
) {

    Card(
        modifier = Modifier
            .fillMaxWidth()
            .padding(horizontal = 20.dp)
            .clickable { onPlay() },
        shape = RoundedCornerShape(12.dp),
        elevation = CardDefaults.cardElevation(3.dp),
        colors = CardDefaults.cardColors(containerColor = Color(0xFFEFF7EE))
    ) {

        Column(Modifier.padding(14.dp)) {

            Row(verticalAlignment = Alignment.CenterVertically) {
                Icon(
                    Icons.Default.VideoFile,
                    contentDescription = null,
                    tint = MaterialTheme.colorScheme.primary,
                    modifier = Modifier.size(34.dp)
                )

                Spacer(Modifier.width(12.dp))

                Text(
                    file.name,
                    fontWeight = FontWeight.Bold,
                    style = MaterialTheme.typography.bodyLarge
                )
            }

            Spacer(Modifier.height(6.dp))

            Text(
                "${formatDuration(FileManagerUtility.getVideoDuration(file))} • " +
                        FileManagerUtility.formatFileSize(file.length()),
                style = MaterialTheme.typography.bodySmall,
                color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.7f),
                modifier = Modifier.padding(start = 46.dp)
            )

            Spacer(Modifier.height(6.dp))
```

```
            Row(modifier = Modifier.padding(start = 46.dp)) {

                Text(
                    "[Edit]",
                    modifier = Modifier.clickable { onEdit() },
                    color = MaterialTheme.colorScheme.primary,
                    style = MaterialTheme.typography.bodySmall
                )

                Spacer(Modifier.width(20.dp))

                Text(
                    "[Delete]",
                    modifier = Modifier.clickable { onDelete() },
                    color = Color.Red,
                    style = MaterialTheme.typography.bodySmall
                )
            }
        }
    }
}
```

5.4 Update file : ui/home/**HomeScreen**.kt

```
package id.antasari.p8_multimedia_nimanda.ui.home

import android.net.Uri
import android.widget.Toast
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.CameraAlt
import androidx.compose.material.icons.filled.Mic
import androidx.compose.material.icons.filled.PlayArrow
import androidx.compose.material.icons.filled.Videocam
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import androidx.navigation.NavController
import id.antasari.p8_multimedia_nimanda.R
import id.antasari.p8_multimedia_nimanda.ui.Screen
```

```
@Composable
fun HomeScreen(
    navController: NavController
) {

    val context = LocalContext.current
    val scrollState = rememberScrollState()

    val gradient = Brush.verticalGradient(
        listOf(Color(0xFF2ECC71), Color(0xFF00A86B))
    )

    Column(
        modifier = Modifier
            .fillMaxSize()
            .background(Color(0xFFF5F7F6))
            .verticalScroll(scrollState)
    ) {

        // ================ HEADER ================
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(200.dp)
                .background(gradient),
            contentAlignment = Alignment.TopCenter
        ) {
            Column(horizontalAlignment = Alignment.CenterHorizontally) {

                Spacer(Modifier.height(18.dp))
                Image(
                    painter = painterResource(id = R.drawable.logo_multimedia),
                    contentDescription = "Logo",
                    modifier = Modifier
                        .size(96.dp)
                        .clip(CircleShape),
                    contentScale = ContentScale.Crop
                )

                Spacer(Modifier.height(12.dp))

                Text(
                    text = "Multimedia Studio",
                    style = MaterialTheme.typography.titleLarge,
                    color = Color.White
                )

                Spacer(Modifier.height(24.dp))
            }
        }
```

```
// ================ HERO CARD ================
Card(
    modifier = Modifier
        .padding(horizontal = 18.dp)
        .height(180.dp)
        .offset(y = (-26).dp),
    shape = RoundedCornerShape(18.dp),
    elevation = CardDefaults.cardElevation(8.dp)
) {
    Image(
        painter = painterResource(id = R.drawable.hero_multimedia),
        contentDescription = "Hero",
        contentScale = ContentScale.Crop
    )
}

Spacer(Modifier.height(8.dp))

// ================ MENU GRID ================
Column(
    modifier = Modifier.padding(horizontal = 18.dp)
) {

    // ===== Row 1 =====
    Row(
        modifier = Modifier.fillMaxWidth(),
        horizontalArrangement = Arrangement.spacedBy(14.dp)
    ) {
        MenuCard(
            icon = Icons.Default.Mic,
            title = "Record Audio",
            modifier = Modifier.weight(1f)
        ) {
            navController.navigate(Screen.AudioRecorder.route)
        }
        MenuCard(
            icon = Icons.Default.PlayArrow,
            title = "Play Audio",
            modifier = Modifier.weight(1f)
        ) {
            navController.navigate(Screen.AudioPlayer.route)
        }
    }

    Spacer(Modifier.height(14.dp))

    // ===== Row 2 =====
    Row(
        modifier = Modifier.fillMaxWidth(),
        horizontalArrangement = Arrangement.spacedBy(14.dp)
    ) {

        MenuCard(
            icon = Icons.Default.Videocam,
            title = "Play Video",
            modifier = Modifier.weight(1f)
        ) {
```

```
                // ===== AUTO-DETECT VIDEO (FINAL FIX) =====
                val videos = context.filesDir
                    .listFiles()
                    ?.filter { it.extension.lowercase() == "mp4" }

                if (!videos.isNullOrEmpty()) {
                    val videoFile = videos.first()
                    val encoded = Uri.encode(videoFile.absolutePath)
                    navController.navigate(
                        Screen.VideoPlayer.passPath(encoded)
                    )
                } else {
                    Toast
                        .makeText(
                            context,
                            "Belum ada video, silakan rekam atau ambil dari gallery",
                            Toast.LENGTH_SHORT
                        )
                        .show()
                }
            }

            MenuCard(
                icon = Icons.Default.CameraAlt,
                title = "Camera & Gallery",
                modifier = Modifier.weight(1f)
            ) {
                navController.navigate(Screen.CameraGallery.route)
            }
        }

        Spacer(Modifier.height(24.dp))
        // ================ FOOTER ================
        val footerStyle = MaterialTheme.typography.bodySmall.copy(
            color = MaterialTheme.colorScheme.onSurface.copy(alpha = 0.75f)
        )

        Column(
            modifier = Modifier.fillMaxWidth(),
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
            Text("Copyright ©2025", style = footerStyle)
            Text(
                "Praktikum #8 Menggunakan Multimedia",
                style = footerStyle.copy(fontWeight = FontWeight.Bold)
            )
            Text("Kuliah Mobile Programming S1 Teknologi Informasi", style = footerStyle)
            Text("UIN Antasari Banjarmasin", style = footerStyle)
        }

        Spacer(Modifier.height(20.dp))
    }
  }
}
```

```
// ================= MENU CARD =================
@Composable
fun MenuCard(
    icon: androidx.compose.ui.graphics.vector.ImageVector,
    title: String,
    modifier: Modifier = Modifier,
    onClick: () -> Unit
) {
    Card(
        modifier = modifier.height(120.dp),
        shape = RoundedCornerShape(16.dp),
        elevation = CardDefaults.cardElevation(6.dp),
        onClick = onClick
    ) {
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(12.dp),
            verticalArrangement = Arrangement.Center,
            horizontalAlignment = Alignment.CenterHorizontally
        ) {

            Box(
                modifier = Modifier
                    .size(54.dp)
                    .clip(CircleShape)
                    .background(Color(0xFF00C06B)),
                contentAlignment = Alignment.Center
            ) {
                Icon(
                    icon,
                    contentDescription = title,
                    tint = Color.White,
                    modifier = Modifier.size(28.dp)
                )
            }

            Spacer(Modifier.height(10.dp))

            Text(title, style = MaterialTheme.typography.bodyMedium)
        }
    }
}
```

5.5 Update file : util/**FileManagerUtility**.kt

```
package id.antasari.p8_multimedia_nimanda.util

import android.content.Context
import android.media.MediaMetadataRetriever
import android.net.Uri
import androidx.media3.common.MediaItem
import androidx.media3.exoplayer.ExoPlayer
import java.io.File
import kotlin.math.roundToInt
```

```
data class AudioFileData(
    val file: File,
    val durationMs: Long,
    val sizeText: String
)

object FileManagerUtility {

    // ==================== AUDIO ====================

    /** Ambil seluruh file audio rekaman (.mp4) dari internal storage */
    fun getAllAudioFiles(context: Context): List<File> {
        return context.filesDir
            .listFiles()
            ?.filter {
                it.extension.lowercase() == "mp4" &&
                        it.name.startsWith("audio_")
            }
            ?.sortedByDescending { it.lastModified() }
            ?: emptyList()
    }
    /** Mengambil durasi audio menggunakan ExoPlayer tanpa memutar */
    fun getAudioDuration(context: Context, file: File): Long {
        return try {
            val retriever = MediaMetadataRetriever()
            retriever.setDataSource(file.absolutePath)
            val duration = retriever
                .extractMetadata(MediaMetadataRetriever.METADATA_KEY_DURATION)
                ?.toLong() ?: 0L
            retriever.release()
            duration
        } catch (e: Exception) {
            0L
        }
    }

    // ==================== VIDEO ====================

    /** Ambil seluruh file video (.mp4) dari internal storage */
    fun getAllVideoFiles(context: Context): List<File> {
        return context.filesDir
            .listFiles()
            ?.filter {
                it.extension.lowercase() == "mp4" &&
                        it.name.startsWith("video_")
            }
            ?.sortedByDescending { it.lastModified() }
            ?: emptyList()
    }
```

```
    /** Mengambil durasi video menggunakan MediaMetadataRetriever */
    fun getVideoDuration(file: File): Long {
        return try {
            val retriever = MediaMetadataRetriever()
            retriever.setDataSource(file.absolutePath)
            val duration =
                retriever.extractMetadata(
                    MediaMetadataRetriever.METADATA_KEY_DURATION
                )?.toLong() ?: 0L
            retriever.release()
            duration
        } catch (e: Exception) {
            0L
        }
    }

    // ==================== COMMON ====================

    /** Convert bytes → KB / MB */
    fun formatFileSize(bytes: Long): String {
        val kb = bytes / 1024f
        return if (kb > 1024)
            "${(kb / 1024).roundToInt()} MB"
        else
            "${kb.roundToInt()} KB"
    }

    /** Rename file */
    fun renameFile(oldFile: File, newName: String): Boolean {
        val newFile = File(oldFile.parent, newName)
        return oldFile.renameTo(newFile)
    }

    /** Delete file */
    fun deleteFile(file: File): Boolean {
        return file.delete()
    }
}
```

5.6 Update file : util/**VideoFileData**.kt

```
package id.antasari.p8_multimedia_nimanda.util

import java.io.File

data class VideoFileData(
    val file: File,
    val durationMs: Long,
    val sizeText: String
)
```

5.7 Update file : ui/**Screen**.kt

```
package id.antasari.p8_multimedia_nimanda.ui

sealed class Screen(val route: String) {
    object Home : Screen("home")
    object AudioRecorder : Screen("audio_recorder")
    object AudioPlayer : Screen("audio_player/{audioPath}") {
        fun passPath(encoded: String): String =
            "audio_player/$encoded"
    }
    object VideoPlayer : Screen("video_player/{videoPath}") {
        fun passPath(encoded: String): String =
            "video_player/$encoded"
    }
    object CameraGallery : Screen("camera_gallery")
}
```

5.8 Update file : ui/**Navgraph**.kt

```
package id.antasari.p8_multimedia_nimanda.ui

import android.net.Uri
import androidx.compose.runtime.Composable
import androidx.navigation.NavType
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import androidx.navigation.compose.rememberNavController
import androidx.navigation.navArgument
import id.antasari.p8_multimedia_nimanda.ui.gallery.CameraGalleryScreen
import id.antasari.p8_multimedia_nimanda.ui.home.HomeScreen
import id.antasari.p8_multimedia_nimanda.ui.player.AudioPlayerScreen
import id.antasari.p8_multimedia_nimanda.ui.recorder.AudioRecorderScreen
import id.antasari.p8_multimedia_nimanda.ui.video.VideoPlayerScreen

@Composable
fun AppNavHost(startDestination: String = Screen.Home.route) {

    val navController = rememberNavController()

    NavHost(
        navController = navController,
        startDestination = startDestination
    ) {

        // ================ HOME ================
        composable(Screen.Home.route) {
            HomeScreen(navController = navController)
        }
```

```
        // ================ AUDIO RECORDER ================
        composable(Screen.AudioRecorder.route) {
            AudioRecorderScreen(
                onBack = { navController.popBackStack() },
                onNavigate = { route ->
                    navController.navigate(route)
                }
            )
        }

        // ================ AUDIO PLAYER ================
        composable(
            route = Screen.AudioPlayer.route,
            arguments = listOf(
                navArgument("audioPath") { type = NavType.StringType }
            )
        ) { backStackEntry ->

            val audioPath = Uri.decode(
                backStackEntry.arguments?.getString("audioPath") ?: ""
            )

            AudioPlayerScreen(
                onBack = { navController.popBackStack() },
                audioPath = audioPath
            )
        }
        // ================ VIDEO PLAYER ================
        composable(
            route = Screen.VideoPlayer.route,
            arguments = listOf(
                navArgument("videoPath") { type = NavType.StringType }
            )
        ) { backStackEntry ->

            val videoPath = Uri.decode(
                backStackEntry.arguments?.getString("videoPath") ?: ""
            )

            VideoPlayerScreen(
                onBack = { navController.popBackStack() },
                videoPath = videoPath
            )
        }

        // ================ CAMERA & GALLERY ================
        composable(Screen.CameraGallery.route) {
            CameraGalleryScreen(
                onBack = { navController.popBackStack() },
                onNavigate = { route ->
                    navController.navigate(route)
                }
            )
        }

    }
}
```

5.9 Jalankan **Build** lalu **Run app** di emulator / device.

1. Jalankan aplikasi pada real device atau emulator yang mendukung kamera.
2. Masuk ke menu Camera & Gallery.
3. Lakukan pengujian berikut:
   - ✓ Ambil foto dari kamera.
   - ✓ Pilih foto / video / audio dari galeri.
   - ✓ Lakukan zoom dan geser pada preview image.
   - ✓ Simpan foto ke Galeri HP.
   - ✓ Putar audio pada Audio Player.
   - ✓ Putar video pada Video Player dan uji zoom + geser video.
4. Uji rotasi layar (portrait ↔ landscape) pada image dan video.

✓ **Checklist**

📷 Camera & Gallery
- ☐ Menu Open Camera dan Choose from Gallery tampil sebagai Card Menu.
- ☐ Icon pada Card Menu tampil besar dan proporsional.
- ☐ Foto hasil kamera tampil pada Image Preview.
- ☐ Foto dapat di-zoom (pinch zoom).
- ☐ Foto dapat di-geser (pan) ke segala arah.
- ☐ Foto tidak hilang saat rotasi layar.
- ☐ Tombol Simpan ke Galeri muncul di bawah Image Preview.
- ☐ Foto tersimpan permanen dan muncul di Galeri HP.

🎵 Audio
- ☐ Daftar rekaman hanya menampilkan file audio.
- ☐ Audio dapat diputar otomatis saat dipilih.
- ☐ Audio Player berjalan normal.

🎬 Video
- ☐ Video dapat diputar di Video Player.
- ☐ Mode fullscreen berfungsi.
- ☐ Video dapat di-zoom.
- ☐ Video dapat di-geser (pan).
- ☐ Kontrol play/pause dan slider tetap berfungsi.

🔄 Gesture & UX
- ☐ Zoom dan pan bekerja bersamaan.
- ☐ Gesture tidak mengganggu scroll layar.
- ☐ Tampilan tetap stabil pada berbagai orientasi layar.