**Wardrobe project code**

**MainActivity.kt:**

package com.example.clothingapp  
  
import android.os.Bundle  
import android.widget.Toast  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.result.ActivityResultLauncher  
import androidx.activity.result.contract.ActivityResultContract  
import androidx.activity.result.contract.ActivityResultContracts  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.tooling.preview.Preview  
import androidx.core.app.ActivityOptionsCompat  
import androidx.navigation.compose.rememberNavController  
import com.example.clothingapp.com.example.clothingapp.AppNavigation  
import com.example.clothingapp.ui.theme.ClothingAppTheme  
  
  
  
  
class MainActivity : ComponentActivity() {  
 private lateinit var cameraLauncher: ActivityResultLauncher<Void?>  
 private lateinit var galleryLauncher: ActivityResultLauncher<String>  
 private lateinit var requestPermissionLauncher: ActivityResultLauncher<String>  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
  
 // Initialize the permission launcher  
 requestPermissionLauncher = registerForActivityResult(ActivityResultContracts.RequestPermission()) **{** isGranted **->** if (!isGranted) {  
 Toast.makeText(this, "Camera permission denied", Toast.LENGTH\_SHORT).show()  
 }  
 **}** // Request camera permission  
 requestPermissionLauncher.launch(android.Manifest.permission.CAMERA)  
  
 // Initialize the camera launcher  
 cameraLauncher = registerForActivityResult(ActivityResultContracts.TakePicturePreview()) **{** bitmap **->** bitmap?.let **{** Toast.makeText(this, "Image captured from Camera", Toast.LENGTH\_SHORT).show()  
 // Handle camera image here  
 **}  
 }** // Initialize the gallery launcher  
 galleryLauncher = registerForActivityResult(ActivityResultContracts.GetContent()) **{** uri **->** uri?.let **{** Toast.makeText(this, "Image selected from Gallery", Toast.LENGTH\_SHORT).show()  
 // Handle gallery image here  
 **}  
 }** // Set the composable content  
 setContent **{** ClothingAppTheme **{** AppNavigation(cameraLauncher = cameraLauncher, galleryLauncher = galleryLauncher)  
 **}  
 }** }  
}  
  
@Preview(showBackground = true)  
@Composable  
fun PreviewApp() {  
 ClothingAppTheme **{** val navController = rememberNavController()  
 // Use mock implementations for preview, as ActivityResultLaunchers can't be used in previews  
 AppNavigation(  
 cameraLauncher = object : ActivityResultLauncher<Void?>() {  
 override fun launch(input: Void?) {}  
 override val contract: ActivityResultContract<Void?, \*>  
 get() = *TODO("Not yet implemented")* override fun launch(input: Void?, options: ActivityOptionsCompat?) {  
 *TODO("Not yet implemented")* }  
  
 override fun unregister() {}  
 fun getContract(): ActivityResultContracts.TakePicturePreview = ActivityResultContracts.TakePicturePreview()  
 },  
 galleryLauncher = object : ActivityResultLauncher<String>() {  
 override fun launch(input: String) {}  
 override val contract: ActivityResultContract<String, \*>  
 get() = *TODO("Not yet implemented")* override fun launch(input: String, options: ActivityOptionsCompat?) {  
 *TODO("Not yet implemented")* }  
  
 override fun unregister() {}  
 fun getContract(): ActivityResultContracts.GetContent = ActivityResultContracts.GetContent()  
 }  
 )  
 **}**}  
  
/\*  
this is working correctly  
@Preview(showBackground = true)  
@Composable  
fun PreviewApp() {  
 ClothingAppTheme{  
 // Replace with a preview of a composable that doesn't depend on launchers  
 ClothingAppScreen(navController = rememberNavController())  
 }  
}  
\*/  
//  
// @Preview(showBackground = true)  
// @Composable  
// fun PreviewApp() {  
// ClothingAppTheme {  
//  
// // AppNavigation(cameraLauncher = cameraLauncher, galleryLauncher = galleryLauncher))  
// // {  
// //}  
// }  
// }

**AppNavigation.kt:**

package com.example.clothingapp.com.example.clothingapp  
  
import androidx.activity.result.ActivityResultLauncher  
import androidx.compose.runtime.Composable  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import com.example.clothingapp.WardrobeScreen  
  
  
@Composable  
fun AppNavigation(  
 cameraLauncher: ActivityResultLauncher<Void?>,  
 galleryLauncher: ActivityResultLauncher<String>  
) {  
 val navController = rememberNavController()  
  
 NavHost(navController = navController, startDestination = "get\_started") **{** *composable*("get\_started") **{** ClothingAppScreen(navController) **}** *composable*("wardrobe") **{** WardrobeScreen(  
 navController = navController,  
 cameraLauncher = cameraLauncher,  
 galleryLauncher = galleryLauncher  
 )  
 **}** *composable*("home") **{** NavigationScreen(navController) **}** *composable*("traditional") **{** TraditionalDressesScreen() **}** *composable*("western") **{** WesternDressesScreen() **}  
 }**}

**NavigationScreen.kt:**

package com.example.clothingapp.com.example.clothingapp  
  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.height  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material3.Button  
import androidx.compose.material3.ButtonDefaults  
import androidx.compose.material3.Scaffold  
import androidx.compose.material3.Text  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavController  
  
  
@Composable  
fun NavigationScreen(navController: NavController)  
{  
  
 Scaffold(  
 topBar = **{** // TopAppBar(title = { Text("Dress Categories") })  
 **}**,  
 modifier = Modifier.*padding*(16.*dp*) // Added padding to the content of Scaffold  
 ) **{** paddingValues **->** Column(modifier = Modifier.*fillMaxSize*().*padding*(paddingValues)) **{** // Use paddingValues here  
 Button(  
 onClick = **{** navController.navigate("traditional") **}**,  
 shape = *RoundedCornerShape*(8.*dp*),  
 colors = ButtonDefaults.buttonColors(containerColor = *Color*(0xFFF1C27D)),  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(vertical = 10.*dp*)  
 .*height*(50.*dp*)  
 ) **{** Text(text = "Traditional Dresses", color = Color.White)  
 **}** Button(  
 onClick = **{** navController.navigate("western") **}**,  
 shape = *RoundedCornerShape*(8.*dp*),  
 colors = ButtonDefaults.buttonColors(containerColor = *Color*(0xFFF1C27D)),  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(vertical = 10.*dp*)  
 .*height*(50.*dp*)  
 ) **{** Text(text = "Western Dresses", color = Color.White)  
 **}  
 }  
 }**}

**ClothingAppScreen.kt:**

package com.example.clothingapp.com.example.clothingapp  
  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Box  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.height  
import androidx.compose.foundation.layout.width  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material3.Button  
import androidx.compose.material3.ButtonDefaults  
import androidx.compose.material3.Text  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavController  
import com.example.clothingapp.R  
  
@Composable  
fun ClothingAppScreen(navController: NavController) {  
 Box(modifier = Modifier.*fillMaxSize*()) **{** // Background Image  
 Image(  
 painter = painterResource(id = R.drawable.*img1*), // Replace with your image resource  
 contentDescription = "Clothing App Background",  
 contentScale = ContentScale.Crop,  
 modifier = Modifier.*fillMaxSize*()  
 )  
  
 // Centered "Get Started" button  
 Column(  
 modifier = Modifier.*fillMaxSize*(),  
 verticalArrangement = Arrangement.Center,  
 horizontalAlignment = Alignment.CenterHorizontally  
 ) **{** Button(  
 onClick = **{** navController.navigate("wardrobe") **}**, // Navigate to Wardrobe screen  
 shape = *RoundedCornerShape*(100),  
 colors = ButtonDefaults.buttonColors(containerColor = *Color*(0xFFF5DEB3)),  
 modifier = Modifier  
 .*height*(50.*dp*)  
 .*width*(150.*dp*)  
 ) **{** Text(text = "Get Started", color = Color.Black)  
 **}  
 }  
 }**}

**WardrobeScreen.kt:**

package com.example.clothingapp  
  
import android.widget.Toast  
import androidx.activity.compose.rememberLauncherForActivityResult  
import androidx.activity.result.ActivityResultLauncher  
import androidx.activity.result.contract.ActivityResultContracts  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.shape.RoundedCornerShape  
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.layout.ContentScale  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavController  
  
@Composable  
fun WardrobeScreen(  
 navController: NavController,  
 cameraLauncher: ActivityResultLauncher<Void?>,  
 galleryLauncher: ActivityResultLauncher<String>  
) {  
 val context = *LocalContext*.current  
 var showSourceDialog by remember **{** *mutableStateOf*(false) **}** var showAddToDialog by remember **{** *mutableStateOf*(false) **}** var selectedImageUri by remember **{** *mutableStateOf*<String?>(null) **}** // Handle Source Selection Dialog  
 if (showSourceDialog) {  
 AlertDialog(  
 onDismissRequest = **{** showSourceDialog = false **}**,  
 title = **{** Text(text = "Select Source") **}**,  
 text = **{** Text("Choose how to add a dress:") **}**,  
 confirmButton = **{** Button(  
 onClick = **{** showSourceDialog = false  
 cameraLauncher.launch(null) // Launch Camera  
 **}** ) **{** Text("Camera")  
 **}  
 }**,  
 dismissButton = **{** Button(  
 onClick = **{** showSourceDialog = false  
 galleryLauncher.launch("image/\*") // Launch Gallery  
 **}** ) **{** Text("Gallery")  
 **}  
 }** )  
 }  
  
 // Handle Add To Dialog  
 if (showAddToDialog && selectedImageUri != null) {  
 AlertDialog(  
 onDismissRequest = **{** showAddToDialog = false **}**,  
 title = **{** Text(text = "Add Image To") **}**,  
 text = **{** Text("Where do you want to add this dress?") **}**,  
 confirmButton = **{** Button(  
 onClick = **{** WesternDressesRepository.addDress(  
 imageUri = selectedImageUri!!,  
 name = "Western Dress",  
 description = "Description",  
 imageRes = *TODO*()  
 )  
 Toast.makeText(context, "Image added to Western category", Toast.*LENGTH\_SHORT*).show()  
 showAddToDialog = false  
 selectedImageUri = null  
 **}** ) **{** Text("Western")  
 **}  
 }**,  
 dismissButton = **{** Button(  
 onClick = **{** TraditionalDressesRepository.addDress(  
 imageUri = selectedImageUri!!,  
 name = "Traditional Dress",  
 description = "Description",  
 imageRes = *TODO*()  
 )  
 Toast.makeText(context, "Image added to Traditional category", Toast.*LENGTH\_SHORT*).show()  
 showAddToDialog = false  
 selectedImageUri = null  
 **}** ) **{** Text("Traditional")  
 **}  
 }** )  
 }  
  
 // Main Layout  
 Box(  
 modifier = Modifier  
 .*fillMaxSize*(),  
 contentAlignment = Alignment.Center,  
  
 ) **{** // Background Image  
 Image(  
 painter = painterResource(id = R.drawable.*img2bg*),  
 contentDescription = "Wardrobe Background",  
 contentScale = ContentScale.Crop,  
 modifier = Modifier.*fillMaxSize*()  
 )  
 Column(  
 verticalArrangement = Arrangement.Center,  
 horizontalAlignment = Alignment.CenterHorizontally,  
 modifier = Modifier.*fillMaxSize*()  
 ) **{** Text(  
 text = "My Wardrobe",  
 style = MaterialTheme.typography.titleLarge,  
 modifier = Modifier.*padding*(bottom = 16.*dp*)  
 )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Button for Traditional Dresses  
 Button(  
 onClick = **{** navController.navigate("traditional") **}**,  
 shape = *RoundedCornerShape*(8.*dp*),  
 colors = ButtonDefaults.buttonColors(containerColor = *Color*(0xFFF5DEB3)),  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(horizontal = 20.*dp*)  
 .*height*(50.*dp*)  
 ) **{** Text(text = "Traditional Dresses", color = Color.Black)  
 **}** Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Button for Western Dresses  
 Button(  
 onClick = **{** navController.navigate("western") **}**,  
 shape = *RoundedCornerShape*(8.*dp*),  
 colors = ButtonDefaults.buttonColors(containerColor = *Color*(0xFFF5DEB3)),  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(horizontal = 20.*dp*)  
 .*height*(50.*dp*)  
 ) **{** Text(text = "Western Dresses", color = Color.Black)  
 **}** Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Button for Adding New Dresses  
 Button(  
 onClick = **{** showSourceDialog = true **}**,  
 shape = *RoundedCornerShape*(8.*dp*),  
 colors = ButtonDefaults.buttonColors(containerColor = *Color*(0xFFF5DEB3)),  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(horizontal = 20.*dp*)  
 .*height*(50.*dp*)  
 ) **{** Text(text = "Add More Dresses", color = Color.Black)  
 **}  
 }  
 }**}  
  
@Composable  
fun rememberGalleryLauncher(onImageSelected: (String) -> Unit): ActivityResultLauncher<String> {  
 return rememberLauncherForActivityResult(  
 contract = ActivityResultContracts.GetContent(),  
 onResult = **{** uri **->** if (uri != null) {  
 onImageSelected(uri.toString()) // Pass the URI as a string  
 }  
 **}** )  
}

**TradationalDressScreen.kt**

package com.example.clothingapp.com.example.clothingapp  
  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.Spacer  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.height  
import androidx.compose.foundation.layout.heightIn  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.lazy.LazyColumn  
import androidx.compose.foundation.lazy.items  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material3.Card  
import androidx.compose.material3.CardDefaults  
import androidx.compose.material3.MaterialTheme  
import androidx.compose.material3.Text  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.unit.dp  
import com.example.clothingapp.TraditionalDressesRepository  
  
@Composable  
fun TraditionalDressesScreen() {  
 val dresses = TraditionalDressesRepository.dresses  
  
 LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(16.*dp*),  
 verticalArrangement = Arrangement.spacedBy(12.*dp*)  
 ) **{** *items*(dresses) **{** dress **->** Card(  
 shape = *RoundedCornerShape*(12.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*),  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** Column(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(16.*dp*)  
 ) **{** Image(  
 painter = painterResource(id = dress.imageRes),  
 contentDescription = dress.name,  
 contentScale = ContentScale.Fit,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*heightIn*(min = 200.*dp*, max = 200.*dp*)  
 )  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = dress.name, style = MaterialTheme.typography.titleMedium)  
 Spacer(modifier = Modifier.*height*(4.*dp*))  
 Text(text = dress.description, style = MaterialTheme.typography.bodyMedium)  
 **}  
 }  
 }  
 }**}

**WesternWardrobeScreen.kt:**

package com.example.clothingapp.com.example.clothingapp  
  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.Spacer  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.height  
import androidx.compose.foundation.layout.heightIn  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.lazy.LazyColumn  
import androidx.compose.foundation.lazy.items  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material3.Card  
import androidx.compose.material3.CardDefaults  
import androidx.compose.material3.MaterialTheme  
import androidx.compose.material3.Text  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.unit.dp  
import com.example.clothingapp.WesternDressesRepository  
  
  
@Composable  
fun WesternDressesScreen() {  
 val dresses = WesternDressesRepository.dresses  
  
 LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(16.*dp*),  
 verticalArrangement = Arrangement.spacedBy(12.*dp*)  
 ) **{** *items*(dresses) **{** dress **->** Card(  
 shape = *RoundedCornerShape*(12.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*),  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** Column(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(16.*dp*)  
 ) **{** Image(  
 painter = painterResource(id = dress.imageRes),  
 contentDescription = dress.name,  
 contentScale = ContentScale.Fit,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*heightIn*(min = 200.*dp*, max = 200.*dp*)  
 )  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = dress.name, style = MaterialTheme.typography.titleMedium)  
 Spacer(modifier = Modifier.*height*(4.*dp*))  
 Text(text = dress.description, style = MaterialTheme.typography.bodyMedium)  
 **}  
 }  
 }  
 }**}

**Dress class:**

package com.example.clothingapp  
  
data class Dress(  
  
 val imageRes: Int, // Resource ID of the image  
 val name: String, // Name of the dress  
 val description: String // Description of the dress  
)

**Repositories**

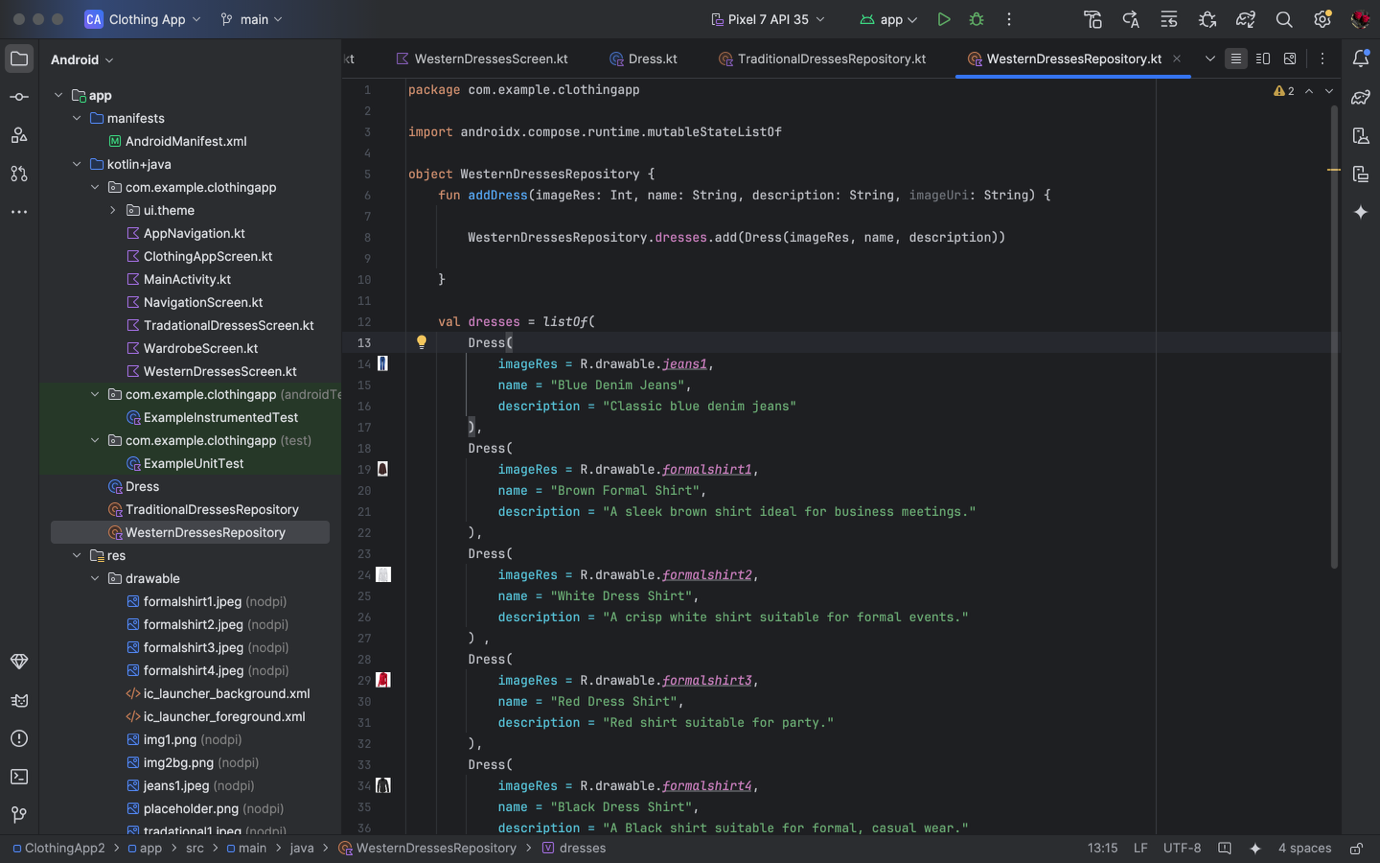
**TradartionalDressRepository.kt:**

package com.example.clothingapp  
  
import androidx.compose.runtime.mutableStateListOf  
  
object TraditionalDressesRepository {  
 val dresses = *mutableStateListOf*<Dress>(  
 Dress(  
 imageRes = R.drawable.*tradational1*,  
 name = "Embroidered Kurta",  
 description = "Embroidered traditional kurta for formal and casual occasions."  
 ),  
 Dress(  
 imageRes = R.drawable.*tradational2*,  
 name = "Black Long Shirt",  
 description = "A vibrant long dress for casual wear."  
 ),  
 Dress(  
 imageRes = R.drawable.*tradational3*,  
 name = "Lehnga",  
 description = "An elegant ubtan lehnga for mehndi event."  
 ),  
 Dress(  
 imageRes = R.drawable.*tradational4*,  
 name = "Kameez",  
 description = "An elegant long Kameez with intricate patterns."  
 ),  
 Dress(  
 imageRes = R.drawable.*tradational5*,  
 name = "Maxi",  
 description = "Mehroon Maxi for formal events."  
 ),  
 Dress(  
 imageRes = R.drawable.*tradational6*,  
 name = "Kurta Trouser",  
 description = "Long seagreen kurta."  
 )  
 )  
  
 fun addDress(imageRes: Int, name: String, description: String, imageUri: String) {  
 dresses.add(Dress(imageRes, name, description))  
 }  
}

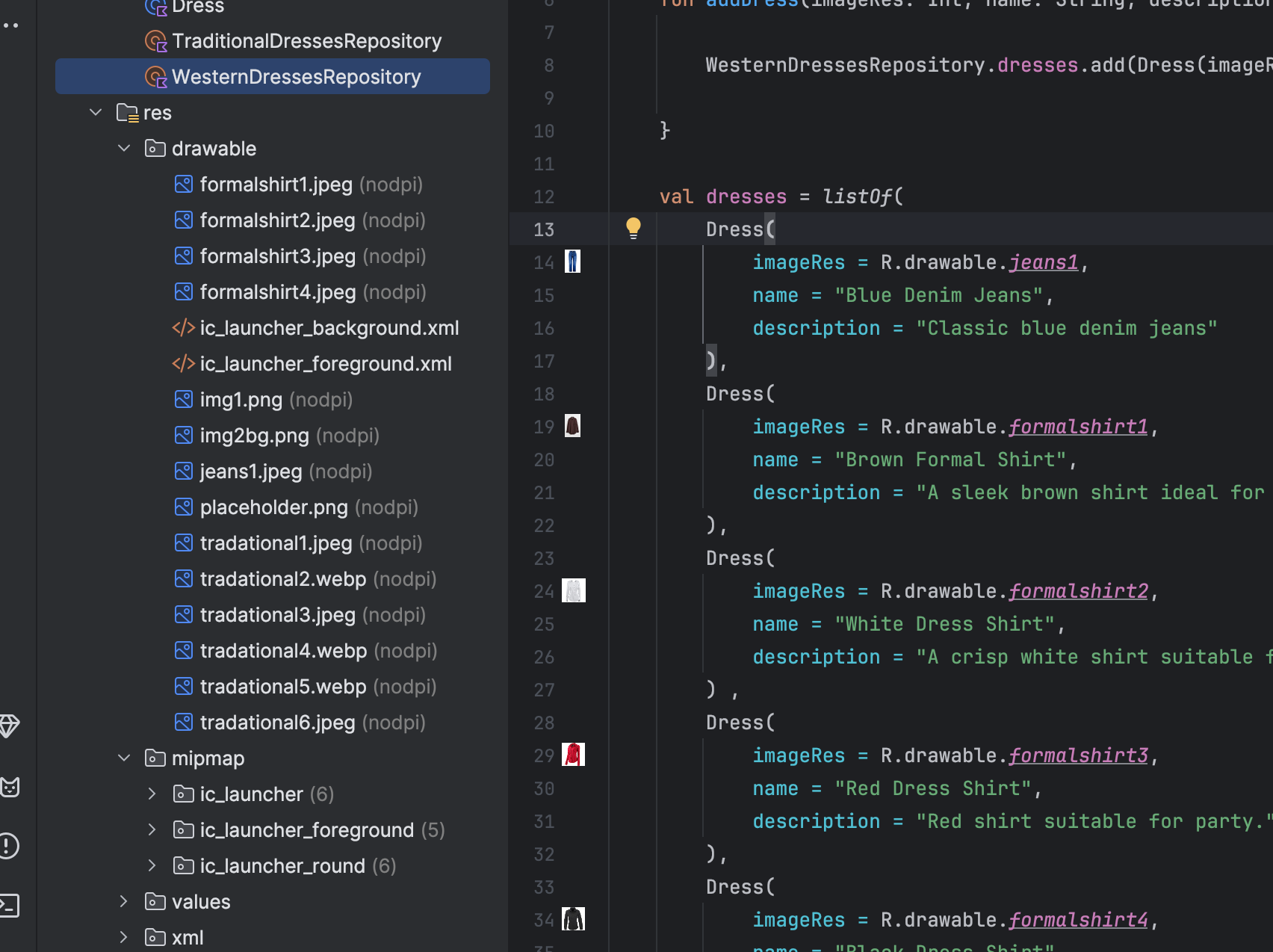
**WesternDressRepository.kt:**

package com.example.clothingapp  
  
import androidx.compose.runtime.mutableStateListOf  
  
object WesternDressesRepository {  
 fun addDress(imageRes: Int, name: String, description: String, imageUri: String) {  
  
 WesternDressesRepository.dresses.add(Dress(imageRes, name, description))  
  
 }  
  
 val dresses = *listOf*(  
 Dress(  
 imageRes = R.drawable.*jeans1*,  
 name = "Blue Denim Jeans",  
 description = "Classic blue denim jeans"  
 ),  
 Dress(  
 imageRes = R.drawable.*formalshirt1*,  
 name = "Brown Formal Shirt",  
 description = "A sleek brown shirt ideal for business meetings."  
 ),  
 Dress(  
 imageRes = R.drawable.*formalshirt2*,  
 name = "White Dress Shirt",  
 description = "A crisp white shirt suitable for formal events."  
 ) ,  
 Dress(  
 imageRes = R.drawable.*formalshirt3*,  
 name = "Red Dress Shirt",  
 description = "Red shirt suitable for party."  
 ),  
 Dress(  
 imageRes = R.drawable.*formalshirt4*,  
 name = "Black Dress Shirt",  
 description = "A Black shirt suitable for formal, casual wear."  
 )  
 // Add more dresses as needed  
 )  
  
 object WesternDressesRepository {  
 val dresses = *mutableStateListOf*<Dress>() // Mutable list to hold dresses  
  
 fun addDress(imageRes: Int, name: String, description: String) {  
 dresses.add(Dress(imageRes, name, description))  
 }  
 }  
  
  
  
}

**Screenshots to locate how files are arranged :**



**Take any Pictures from internet and add them to your resource/ drawable file:**



**LOGO:**

