Step 1: Set Up Your Project

1. Create a new Android project in Android Studio.
2. Open the build.gradle (Module: app) file and add the following dependencies:

implementation("androidx.compose.material:material-icons-extended:1.6.0")

implementation(platform("androidx.compose:compose-bom:2023.08.00"))

Step 2: Create Drawable Resources

Add the required drawable resources (e.g., message.png, ic_lock.png, profile.png) to the res/drawable folder.

Step 3: Implement Composable Components

Copy and paste the provided code for TextNormal, TextBold, PasswordMytextfield, Mytextfield, Buttoncomponent, and ClickableLoginTextComponent into your project.

Ensure that you've imported the necessary Compose libraries.

TextNormal:

```
@Composable
fun TextNormal(value: String) {
   Text(
      text = value,
      modifier = Modifier
         .fillMaxWidth()
         .heightIn(min = 40.dp),
      style = TextStyle(
         color = Color.Black,
         fontSize = 24.sp,
         fontWeight = FontWeight.Normal,
         fontStyle = FontStyle.Normal
      ),
      textAlign = TextAlign.Center,
   )
}
```

TextBold:

```
@Composable
fun TextBold(value: String) {
   Text(
      text = value,
      modifier = Modifier
         .fillMaxWidth()
         .heightIn(),
      style = TextStyle(
         fontSize = 30.sp,
         color = Color.Black,
```

```
            fontWeight = FontWeight.Bold,
            fontStyle = FontStyle.Normal

        ),
        textAlign = TextAlign.Center,
    )
}
```

PasswordMyTextField:

```
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun PasswordMytextfield(labelvalue: String, painterResource: Painter) {
    val password = remember {
        mutableStateOf("")
    }
    val passwordvisble = remember {
        mutableStateOf(false)
    }
    OutlinedTextField(
        modifier = Modifier
            .fillMaxWidth()
            .background(color = Color.White),
        label = { Text(text = labelvalue) },
        value = password.value,
        keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Password),
        onValueChange = {
            password.value = it
        },
        leadingIcon = {
            Icon(painter = painterResource, contentDescription = "")
        },
        trailingIcon = {
            val icon = if (passwordvisble.value) {
                Icons.Filled.Visibility
            } else {
                Icons.Filled.VisibilityOff
            }
            IconButton(onClick = {
                passwordvisble.value = !passwordvisble.value
            }) {
                Icon(imageVector = icon, contentDescription = "")
            }
        },
        visualTransformation = if (passwordvisble.value)
            VisualTransformation.None else PasswordVisualTransformation()
    )
}
```

MyTextField:

```
@Composable
```

```
fun Mytextfield(labelvalue: String, painterResource: Painter) {
    val textvalue = remember {
        mutableStateOf("")
    }
    OutlinedTextField(
        modifier = Modifier
            .fillMaxWidth()
            .background(color = Color.White),
        label = { Text(text = labelvalue) },
        value = textvalue.value,
        keyboardOptions = KeyboardOptions.Default,
        onValueChange = {
            textvalue.value = it
        },
        leadingIcon = {
            Icon(painter = painterResource, contentDescription = "")
        }
    )
}
```

ButtonComponent:

```
@Composable
fun Buttoncomponent(value: String, onclick: () -> Unit) {
    Button(
        onClick = { onclick() },
        modifier = Modifier
            .fillMaxWidth()
            .heightIn(48.dp),
        contentPadding = PaddingValues(0.dp),
        shape = RoundedCornerShape(50.dp),
        colors = ButtonDefaults.buttonColors(Color.Transparent)
    ) {
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .heightIn(48.dp)
                .background(
                    Brush.horizontalGradient(
                        colors = listOf(Color.DarkGray, Color.LightGray),

                        ), shape = RoundedCornerShape(50.dp)
                ),
            contentAlignment = Alignment.Center,
        ) {
            Text(text = value, fontWeight = FontWeight.Bold, fontSize = 18.sp)
        }
    }
}
```

ClickableLoginTextComponent:

```
@Composable
fun ClickableLoginTextComponent(tryingToLogin: Boolean = true, onTextSelected: (String) -> Unit) {
    val initialText =
        if (tryingToLogin) "Already have an account? " else "Don’t have an account yet? "
    val loginText = if (tryingToLogin) "Login" else "Register"

    val annotatedString = buildAnnotatedString {
        append(initialText)
        withStyle(style = SpanStyle(color = Color.LightGray)) {
            pushStringAnnotation(tag = loginText, annotation = loginText)
            append(loginText)
        }
    }

    ClickableText(
        modifier = Modifier
            .fillMaxWidth()
            .heightIn(min = 40.dp),
        style = TextStyle(
            fontSize = 21.sp,
            fontWeight = FontWeight.Normal,
            fontStyle = FontStyle.Normal,
            textAlign = TextAlign.Center
        ),
        text = annotatedString,
        onClick = { offset ->

            annotatedString.getStringAnnotations(offset, offset)
                .firstOrNull()?.also { span ->
                    Log.d("ClickableTextComponent", "{${span.item}}")

                    if (span.item == loginText) {
                        onTextSelected(span.item)
                    }
                }
        },
    )
}
```

Step 4: Create Screens

Copy and paste the provided code for LoginScreen and SignUpScreen.
Make sure to replace the R.drawable.message, R.drawable.ic_lock, and R.drawable.profile references with the actual names of your drawable resources.

LoginScreen:

```
@Composable
```

```
fun LoginScreen(onClick: () -> Unit) {
    Surface(
        color = Color.White,
        modifier = Modifier
            .fillMaxSize()
            .background(Color.White)
            .padding(20.dp),
    ) {
        Column() {
            TextNormal("Hello")
            TextBold("Welcome")
            Mytextfield(
                labelvalue = "Email",
                painterResource(id = R.drawable.message)
            )
            Spacer(modifier = Modifier.height(10.dp))
            PasswordMytextfield(
                labelvalue = "Password",
                painterResource = painterResource(id = R.drawable.ic_lock)
            )
            Spacer(modifier = Modifier.height(90.dp))
            Buttoncomponent(
                value = "Login",
                {}
            )
            ClickableLoginTextComponent(tryingToLogin = false, onTextSelected = {
                onClick()
            })
        }
    }
}

@Preview
@Composable
fun defaultpreviewLoginscreen() {
    LoginScreen(onClick = {})
}
```

SignUpScreen:

```
@Composable
fun SignUpScreen(onclick: () -> Unit) {
    Surface(

        color = Color.White,
        modifier = Modifier
            .fillMaxSize()
            .background(Color.White)
            .padding(20.dp),

        ) {
```

```
        Column(Modifier.fillMaxSize()) {
            TextNormal(value = "Hello")
            TextBold(value = "Create Account")
            Spacer(modifier = Modifier.height(20.dp))
            Mytextfield(
                labelvalue = "First Name",
                painterResource(id = R.drawable.profile)
            )
            Spacer(modifier = Modifier.height(10.dp))
            Mytextfield(
                labelvalue = "Last Name",
                painterResource = painterResource(id = R.drawable.profile)
            )
            Spacer(modifier = Modifier.height(10.dp))
            Mytextfield(
                labelvalue = "Email",
                painterResource = painterResource(id = R.drawable.message)
            )
            Spacer(modifier = Modifier.height(10.dp))
            PasswordMytextfield(
                labelvalue = "Password",
                painterResource = painterResource(id = R.drawable.ic_lock)
            )
            Spacer(modifier = Modifier.height(90.dp))
            Buttoncomponent(
                value = "Register"
            ) {

            }
            ClickableLoginTextComponent(tryingToLogin = true, onTextSelected = {
                onclick()
            })
        }

    }
}

@Preview
@Composable
fun DefaultpreviewSignUpScreen() {
    SignUpScreen(onclick = {})
}
```

Step 5: MainActivity

Copy and paste the provided code for MainActivity.
Make sure the SignUpScreen is being used, and replace the MaterialTheme.colorScheme.background with your desired background color.

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
        super.onCreate(savedInstanceState)
        setContent {

            // A surface container using the 'background' color from the theme
            Surface(
                modifier = Modifier.fillMaxSize(),
                color = MaterialTheme.colorScheme.background
            ) {
                SignUpScreen {

                }
            }
        }
    }
}
```

Step 6: Run Your App Build and run your project on an emulator or a physical device.

Now, you should have a simple authentication screen with login and signup components.
The LoginScreen and SignUpScreen Composables are displayed in the MainActivity.
You can further customize the UI, handle button clicks, and integrate actual authentication logic based on your requirements.