**Nome:** Mariana dos Santos Moreira

**🟦 MainActivity.kt**

package com.example.appcadastro

Define em qual pacote (ou "pasta lógica") o arquivo está. Isso organiza o projeto e separa os arquivos por função.

**class MainActivity : ComponentActivity()**

Essa é a classe principal do app. Ela representa a tela inicial. Ao herdar de ComponentActivity, conseguimos usar Jetpack Compose para construir a interface de forma moderna e declarativa.

**override fun onCreate(savedInstanceState: Bundle?)**

Esse método é executado assim que a tela é criada. O savedInstanceState serve para restaurar informações, caso o usuário rotacione a tela ou o app seja fechado e reaberto.

**super.onCreate(savedInstanceState)**

Chama a versão do método da classe pai. É essencial para que o ciclo de vida da Activity funcione corretamente.

**enableEdgeToEdge()**

Ativa o modo de exibição "borda a borda", deixando o conteúdo ocupar toda a tela, até atrás das barras do sistema.

**setContent** { ... }

Inicia a construção da interface com Jetpack Compose, onde tudo é criado por código de forma visual.

**AppCadastroTheme** { ... }

Aplica o tema personalizado do aplicativo (cores, tipografia, etc.), definido em Theme.kt.

**Scaffold(...) { appPreview() }**

Scaffold é uma estrutura base do Material Design que ajuda a organizar a tela com áreas como topo, corpo e botão flutuante. Dentro dele, a função appPreview() é chamada para exibir a interface principal.

🟩 **ProdutoItem()**

@Composable fun ProdutoItem()

Função responsável por montar a tela de cadastro usando Compose. O @Composable permite que ela desenhe elementos visuais.

🔹 1. **Variáveis de Estado**

var nome by remember { mutableStateOf("") }

Cria uma variável observável, que guarda o valor digitado no campo de texto. Quando o valor muda, a interface é atualizada automaticamente. Esse padrão se repete para telefone, curso, serie e observacao.

🔹 2. **Estrutura com Column**

Column(...)

Organiza os elementos na vertical (um embaixo do outro). Usa fillMaxSize() para ocupar toda a tela e verticalScroll para permitir rolagem se o conteúdo ultrapassar o tamanho visível.

🔹 3. **Cabeçalho com gradiente**

Box(

modifier = Modifier

.height(180.dp)

.background(brush = Brush.horizontalGradient(...))

.fillMaxWidth()

)

Essa Box forma o topo da tela com um fundo colorido em gradiente horizontal. É usada para criar destaque visual no cabeçalho.

🔹 4. **Imagem (Avatar)**

Image(...)

Mostra a imagem de perfil (avatar). Ela é redonda, centralizada, e deslocada verticalmente para sobrepor o cabeçalho de forma estilosa.

🔹 5. **Título**

Text("Cadastre-se", ...)

Exibe o título da tela com fonte seminegrito e tamanho grande. Fica centralizado dentro da Column.

🔹 6. **Campos de Texto**

Cada campo segue esse padrão:

Text("Nome:")

Spacer(...)

TextField(

value = nome,

onValueChange = { nome = it },

label = { Text("Digite seu nome completo") },

...

)

* Text: rótulo acima do campo.
* Spacer: adiciona espaço entre os elementos.
* TextField: campo editável para digitar informações.

Os outros campos (Telefone, Curso, Série e Observação) seguem a mesma estrutura, mudando apenas os rótulos e comportamentos (como teclado numérico ou número de linhas).

🔹 7. **Botão "Cadastrar"**

Button(onClick = { ... }) {

Text("Cadastrar")

}

Botão centralizado que, ao ser clicado, limpa todos os campos. No futuro, você pode substituir essa ação por uma função que salve os dados.

🔹 8. **appPreview()**

@Preview

@Composable

fun appPreview() { ... }

Permite ver uma prévia da interface direto no Android Studio, sem precisar rodar o app no celular. Usa Surface como base para aplicar cor de fundo do tema.

**🟨 Color.kt**

Define as cores usadas no tema do aplicativo.

val Purple80 = Color(0xFFD0BCFF)

val Darkblue = Color(0xff2e4566)

As cores são separadas para tema escuro, tema claro e personalizações. As cores Darkblue e Lightblue são usadas no gradiente do cabeçalho.

**🟧 Theme.kt**

Responsável por aplicar os temas claros e escuros no app.

darkColorScheme(...) // define as cores do modo escuro

lightColorScheme(...) // define as cores do modo claro

A função AppCadastroTheme aplica automaticamente o tema certo de acordo com o sistema do celular. Se for Android 12+, também pode usar cores dinâmicas (baseadas no papel de parede do usuário).

**🟥 Type.kt**

Define a aparência dos textos do app.

val Typography = Typography(

bodyLarge = TextStyle(

fontSize = 16.sp,

fontWeight = FontWeight.Normal,

lineHeight = 24.sp

)

)

Essas configurações são aplicadas nos textos por padrão com o MaterialTheme.

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto. Interface gráfica do usuário

O conteúdo gerado por IA pode estar incorreto.