

No **Passo 7**, vamos implementar a **integração com métodos de pagamento**. A ideia é que o usuário possa finalizar a compra utilizando um gateway de pagamento. Para esta implementação, vamos fazer uma simulação de integração utilizando a **API do PayPal**, uma das opções mais comuns para pagamento online.

Passo 7: Integração com Pagamentos (Simulação com PayPal)

7.1 Criando uma Conta no PayPal Developer

Antes de começar a implementar a integração, você precisa de uma conta de desenvolvedor do PayPal para gerar as credenciais de API.

1. Crie uma conta no PayPal Developer:

- Acesse PayPal Developer e crie uma conta.
- Uma vez logado, navegue até a seção **Dashboard** e clique em **Create App** para criar uma aplicação de sandbox que você usará para testar.

2. Obtenha suas credenciais:


- Após criar a aplicação, você terá um **Client ID** e um **Client Secret**, que serão usados para fazer chamadas à API do PayPal.

7.2 Adicionando Dependências do SDK PayPal

Para integrar o PayPal ao nosso aplicativo, precisamos adicionar as dependências necessárias. Neste exemplo, vamos usar o **PayPal Checkout SDK**.

1. Abra o arquivo `build.gradle` (Module: `app`) e adicione a dependência do SDK do PayPal:

gradle

 Copiar código

```
dependencies { // SDK do PayPal para Android (Checkout) implementation 'com.paypal.sdk:paypalcheckout:0.7.3' }
```

2. Sincronize o projeto:

- Após adicionar a dependência, clique em **"Sync Now"** para garantir que o SDK do PayPal seja adicionado corretamente ao projeto.

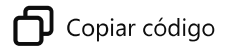
7.3 Configurando o PayPal SDK no Projeto

Agora precisamos inicializar o SDK do PayPal no projeto para que possamos processar pagamentos.

1. Inicialize o SDK do PayPal no MainActivity.kt :

No onCreate() da MainActivity, adicione o seguinte código para inicializar o PayPal SDK com seu Client ID.

kotlin



```
import com.paypal.checkout.PayPalCheckout import com.paypal.checkout.config.CheckoutConfig import
com.paypal.checkout.config.Environment import com.paypal.checkout.config.CurrencyCode import
com.paypal.checkout.config.UserAction class MainActivity : AppCompatActivity() { override fun onCreate(savedInstanceState:
Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_main) // Configurando o PayPal SDK
PayPalCheckout.setConfig( CheckoutConfig( application = this, clientId = "YOUR_PAYPAL_CLIENT_ID", // Insira seu Client ID do
PayPal aqui environment = Environment.SANDBOX, // Usando ambiente sandbox para testes returnUrl =
"com.example.acaiStore://paypalpay", currencyCode = CurrencyCode.USD, userAction = UserAction.PAY_NOW ) ) } }
```

Substitua "YOUR_PAYPAL_CLIENT_ID" pelo seu Client ID obtido no PayPal Developer.

7.4 Configurando o Pagamento na CartActivity

Agora que o SDK do PayPal está configurado, vamos permitir que o usuário faça o pagamento ao finalizar a compra. Para isso, vamos adicionar o código para iniciar o fluxo de pagamento quando o botão "Finalizar Compra" for clicado.

1. Atualize o código da CartActivity.kt :

kotlin



```
import com.paypal.checkout.createorder.CreateOrderActions import com.paypal.checkout.createorder.CreateOrderRequest import
com.paypal.checkout.order.OrderIntent import com.paypal.checkout.paymentbutton.PayPalButton import
com.paypal.checkout.paymentbutton.PayPalButton.OnClickListener class CartActivity : AppCompatActivity() { private lateinit var
recyclerView: RecyclerView private lateinit var totalPriceTextView: TextView private lateinit var checkoutButton: PayPalButton
override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_cart)
recyclerView = findViewById(R.id.cartRecyclerView) totalPriceTextView = findViewById(R.id.totalPrice) checkoutButton =
findViewById(R.id.payPalButton) // Carregar os itens do carrinho CartManager.loadCartItems(this) { cartItems ->
setupRecyclerView(cartItems) updateTotalPrice(cartItems) } // Configurando a ação de pagamento com PayPal checkoutButton.setup(
object : OnClickListener { override fun onClick(actions: CreateOrderActions) { actions.create( CreateOrderRequest.Builder()
```

```
.intent(OrderIntent.CAPTURE) .purchaseUnits( listOf( // Configurando o pedido com o valor total CreateOrderRequest.PurchaseUnit(
amount = CreateOrderRequest.Amount( currencyCode = CurrencyCode.USD, value = CartManager.getTotalPrice().toString() // Valor
total ) ) ) ) .build() ) } } } } private fun setupRecyclerView(cartItems: List<CartItem>) { recyclerView.layoutManager =
LinearLayoutManager(this) recyclerView.adapter = CartAdapter(cartItems) } private fun updateTotalPrice(cartItems: List<CartItem>)
{ val totalPrice = CartManager.getTotalPrice() totalPriceTextView.text = "Total: R$ $totalPrice" } }
```

Aqui, estamos configurando o botão de pagamento do PayPal para iniciar o processo de pagamento com o valor total dos itens no carrinho. Ao clicar em "Finalizar Compra", o fluxo de pagamento é iniciado e o PayPal processa a transação.

7.5 Atualizando o Layout para o Botão do PayPal

Agora, precisamos atualizar o layout do carrinho para incluir o botão de pagamento do PayPal.

1. **Abra o arquivo** `activity_cart.xml` e adicione o botão do PayPal:

xml



```
<?xml version="1.0" encoding="utf-8"?> <androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android" xmlns:app="http://schemas.android.com/apk/res-auto"
android:layout_width="match_parent" android:layout_height="match_parent" android:padding="16dp"> <!-- RecyclerView para exibir os
itens do carrinho --> <androidx.recyclerview.widget.RecyclerView android:id="@+id/cartRecyclerView" android:layout_width="0dp"
android:layout_height="0dp" app:layout_constraintTop_toTopOf="parent"
app:layout_constraintBottom_toTopOf="@+id/totalPriceContainer" app:layout_constraintStart_toStartOf="parent"
app:layout_constraintEnd_toEndOf="parent" android:padding="16dp" /> <!-- Contêiner para o preço total e botão de finalizar -->
<LinearLayout android:id="@+id/totalPriceContainer" android:layout_width="0dp" android:layout_height="wrap_content"
android:orientation="horizontal" android:gravity="center_vertical" app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintStart_toStartOf="parent" app:layout_constraintEnd_toEndOf="parent" android:padding="16dp"> <!-- Preço Total
--> <TextView android:id="@+id/totalPrice" android:layout_width="wrap_content" android:layout_height="wrap_content"
android:text="Total: R$ 0,00" android:textSize="20sp" android:textStyle="bold" android:layout_marginEnd="16dp" /> <!-- Botão de
pagamento do PayPal --> <com.paypal.checkout.paymentbutton.PayPalButton android:id="@+id/payPalButton"
android:layout_width="wrap_content" android:layout_height="wrap_content" android:layout_marginTop="16dp"
android:layout_marginBottom="16dp" app:layout_constraintTop_toBottomOf="@+id/cartRecyclerView"
```

```
app:layout_constraintStart_toStartOf="parent" app:layout_constraintEnd_toEndOf="parent" /> </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Aqui, adicionamos o `PayPalButton` à parte inferior da tela do carrinho, logo abaixo do total da compra.

7.6 Finalizando o Fluxo de Pagamento

Quando o usuário clicar no botão do PayPal, o PayPal gerencia o fluxo de pagamento. Assim que a transação é concluída com sucesso, o PayPal chamará um retorno para finalizar a compra.

1. Gerenciando o retorno do PayPal:

- O SDK do PayPal gerencia a transação e chama automaticamente a função de callback assim que o pagamento for bem-sucedido.

Testando o Pagamento

Agora que a integração com o PayPal foi implementada, você pode testar o fluxo completo de pagamento:

1. Execute o aplicativo no emulador ou dispositivo físico.
2. Adicione itens ao carrinho.
3. Navegue até a tela do carrinho e veja o valor total.
4. Clique no botão do PayPal para iniciar o pagamento.
5. Conclua a transação no ambiente de sandbox do PayPal.

Resumo

Com isso, a integração com o PayPal foi concluída, permitindo que os usuários realizem o pagamento no seu aplicativo de venda de açaí.