

Aqui está um roteiro passo a passo para criar o aplicativo **Tech Burguer** no Android Studio, usando Kotlin para o backend e XML para o frontend.


## 1. Configurar o Projeto

- **Passo 1.1:** Abra o Android Studio e selecione "New Project".
- **Passo 1.2:** Defina o nome do aplicativo como "Tech Burguer", escolha "Empty Activity" e configure a linguagem como **Kotlin**.
- **Passo 1.3:** Defina as versões mínimas e alvo do SDK para garantir compatibilidade com a maioria dos dispositivos.
- **Passo 1.4:** Finalize a configuração do projeto.

## 2. Estruturar a Interface Principal (MainActivity)

- **Passo 2.1:** No arquivo `activity_main.xml`, defina o layout básico.
  - Use **LinearLayout** ou **ConstraintLayout** para estruturar a tela.
  - Adicione um **ImageView** para o logotipo da Tech Burguer.
  - Adicione um **RecyclerView** para listar os itens do menu de hambúrgueres.
  - Adicione botões como "Finalizar Pedido" e "Ver Carrinho".
  - Exemplo:

xml


 Copiar código

```
<LinearLayout android:layout_width="match_parent"
android:layout_height="match_parent" android:orientation="vertical">
<ImageView android:id="@+id/logo" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:src="@drawable/logo" />
<androidx.recyclerview.widget.RecyclerView android:id="@+id/menuRecyclerView"
android:layout_width="match_parent" android:layout_height="0dp"
android:layout_weight="1" /> <Button android:id="@+id/viewCartButton"
android:layout_width="wrap_content" android:layout_height="wrap_content"
android:text="Ver Carrinho" /> <Button android:id="@+id/checkoutButton"
android:layout_width="wrap_content" android:layout_height="wrap_content"
android:text="Finalizar Pedido" /> </LinearLayout>
```

## 3. Criar o Modelo de Dados (Burger e Pedido)

- **Passo 3.1:** Crie uma classe **Burger** no Kotlin para representar os hambúrgueres.

kotlin

 Copiar código

```
data class Burger(val name: String, val description: String, val price: Double,
val imageResId: Int)
```


- **Passo 3.2:** Crie uma classe **Order** para representar o pedido do cliente.

```
data class Order(val burger: Burger, var quantity: Int)
```

## 4. Implementar o Adaptador do RecyclerView

- **Passo 4.1:** Crie um adaptador `BurgerAdapter` para o `RecyclerView`.
  - O adaptador será responsável por exibir cada item do menu.
  - Utilize o layout customizado para cada item de hambúrguer (por exemplo, `burger_item.xml`).
  - Exemplo de um item:

xml

 Copiar código

```
<LinearLayout android:layout_width="match_parent"
    android:layout_height="wrap_content" android:orientation="horizontal">
    <ImageView android:id="@+id/burgerImage" android:layout_width="100dp"
        android:layout_height="100dp" /> <TextView android:id="@+id/burgerName"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:text="Nome do Burguer" /> <TextView
        android:id="@+id/burgerDescription" android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:text="Descrição do Burguer" />
    <TextView android:id="@+id/burgerPrice" android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:text="Preço" /> </LinearLayout>
```

- **Passo 4.2:** Implemente as funções do `BurgerAdapter` no Kotlin.

kotlin

 Copiar código

```
class BurgerAdapter(private val burgerList: List<Burger>, private val
    clickListener: (Burger) -> Unit) :
    RecyclerView.Adapter<BurgerAdapter.BurgerViewHolder>() { class
    BurgerViewHolder(val view: View) : RecyclerView.ViewHolder(view) { val name =
        view.findViewById<TextView>(R.id.burgerName) val description =
        view.findViewById<TextView>(R.id.burgerDescription) val price =
        view.findViewById<TextView>(R.id.burgerPrice) val image =
        view.findViewById<ImageView>(R.id.burgerImage) } override fun
    onCreateViewHolder(parent: ViewGroup, viewType: Int): BurgerViewHolder { val view
        = LayoutInflater.from(parent.context).inflate(R.layout.burger_item, parent, false)
    return BurgerViewHolder(view) } override fun onBindViewHolder(holder:
    BurgerViewHolder, position: Int) { val burger = burgerList[position]
    holder.name.text = burger.name holder.description.text = burger.description
    holder.price.text = "R$ ${burger.price}"
    holder.image.setImageResource(burger.imageResId)
    holder.itemView.setOnClickListener { clickListener(burger) } } override fun
    getItemCount() = burgerList.size }
```

## 5. Implementar as Funcionalidades de Carrinho e Finalização de Pedido

- **Passo 5.1:** Crie uma Activity chamada **CartActivity** para exibir o carrinho.
  - No arquivo `CartActivity.kt`, crie uma lista de pedidos e um RecyclerView semelhante ao menu.
  - Exiba os hambúrgueres adicionados ao carrinho e permita ao usuário alterar as quantidades.
- **Passo 5.2:** Adicione funcionalidade ao botão "Finalizar Pedido" que navega para uma tela de resumo do pedido.

## 6. Adicionar Funcionalidades Avançadas (Opcional)

- **Passo 6.1:** Integre uma API de pagamento, como o **PagSeguro** ou **Stripe**, para processar pagamentos.
- **Passo 6.2:** Adicione a funcionalidade de login e cadastro de usuários com autenticação Firebase.
- **Passo 6.3:** Implemente uma API para pedidos em tempo real (Firebase Realtime Database ou Firestore).
- **Passo 6.4:** Adicione notificações push para informar sobre promoções ou status de pedidos.

## 7. Testar o Aplicativo

- **Passo 7.1:** Execute o app no emulador do Android Studio para testar a usabilidade.
- **Passo 7.2:** Teste a aplicação em diferentes tamanhos de tela para garantir a responsividade.

## 8. Publicar o Aplicativo

- **Passo 8.1:** Gere o arquivo APK ou AAB final.
- **Passo 8.2:** Publique o aplicativo na Google Play Store seguindo os requisitos de publicação.

Esse roteiro cobre a criação do aplicativo **Tech Burger** com as funcionalidades principais de menu, carrinho e finalização de pedido.