Seção 12: Protótipos Visuais e Material Design 3

Desenvolvimento de Aplicativos Móveis - Flutter

Introdução

Uma interface de autenticação bem projetada não é apenas bonita - ela inspira confiança, reduz fricção e melhora significativamente a taxa de conversão. Nesta seção, vamos mergulhar em:

- **Princípios de design** específicos para fluxos de autenticação
- **Material Design 3** e como aplicá-lo consistentemente
- **Especificações detalhadas** de cada tela (cores, fontes, espaçamentos)
- **Componentes reutilizáveis** prontos para implementação
- **Prompts para IA** que você pode usar para gerar protótipos visuais

Ao final desta seção, você terá tudo que precisa para criar interfaces de autenticação profissionais e modernas, seja codificando do zero ou usando ferramentas de IA para acelerar o processo.

Parte 1: Princípios de Design para Autenticação

1.1. Psicologia e Confiança

Autenticação é um momento crítico onde o usuário está confiando informações sensíveis ao seu app. O design precisa transmitir:

Segurança visual:

- Use ícones de cadeado e escudo sutilmente
- Cores profissionais (evite cores muito vibrantes na tela de login)
- Espaçamento generoso (não pareça apressado ou "cramped")
- Tipografia limpa e legível

Simplicidade:

- Máximo 1-2 ações principais por tela
- Hierarquia visual clara (o que fazer primeiro é óbvio)
- Progressão linear (um passo de cada vez)

Feedback visual imediato:

- Validação em tempo real (mas não agressiva)
- Estados de loading claros
- Mensagens de erro empáticas, não acusatórias

1.2. Hierarquia Visual

Em uma tela de autenticação, a hierarquia deve ser:

1. **Título/Logo** (onde estou?)
2. **Campos de entrada** (o que preciso fazer?)
3. **Botão primário** (ação principal - ex: "Entrar")
4. **Ações secundárias** (esqueci senha, cadastrar-se)
5. **Alternativas** (login social)

Regra de ouro: O usuário deve saber onde olhar e o que fazer em menos de 2 segundos.

1.3. Densidade e Respiração

- **Padding externo:** Mínimo 16-24px nas laterais
- **Espaçamento entre campos:** 12-16px
- **Espaçamento entre seções:** 24-32px
- **Altura de campos:** 48-56px (alvo de toque confortável)
- **Tamanho mínimo de botão:** 48px altura

1.4. Cores e Contraste

Para autenticação:

- Background: Neutro claro (`#FAFAFA`, `#FFFFFF`) ou escuro (`#121212`)
- Primária: Cor da marca (mas não exagerada)
- Campos: Fundo sutil (`#F5F5F5` ou outline)
- Erros: Vermelho (`#D32F2F` ou `#F44336`)
- Sucesso: Verde (`#388E3C` ou `#4CAF50`)

Contraste mínimo (WCAG AA):

- Texto normal: 4.5:1
- Texto grande: 3:1
- Elementos interativos: 3:1

Parte 2: Sistema de Design - Material Design 3

2.1. Cores (Color Scheme)

Material Design 3 usa um sistema de cores baseado em "color roles". Aqui está um esquema completo para autenticação:

dart

```dart
// lib/theme/app_colors.dart

import 'package:flutter/material.dart';

class AppColors {
  // ===== LIGHT THEME =====

  // Primary (cor principal da marca)
  static const Color primaryLight = Color(0xFF6750A4); // Roxo vibrante
  static const Color onPrimaryLight = Color(0xFFFFFFFF); // Texto sobre primária
  static const Color primaryContainerLight = Color(0xFFEADDFF); // Container primário
  static const Color onPrimaryContainerLight = Color(0xFF21005D); // Texto no container

  // Secondary (ações secundárias)
  static const Color secondaryLight = Color(0xFF625B71);
  static const Color onSecondaryLight = Color(0xFFFFFFFF);
  static const Color secondaryContainerLight = Color(0xFFE8DEF8);
  static const Color onSecondaryContainerLight = Color(0xFF1D192B);

  // Surface (backgrounds)
  static const Color surfaceLight = Color(0xFFFEF7FF);
  static const Color onSurfaceLight = Color(0xFF1D1B20);
  static const Color surfaceVariantLight = Color(0xFFE7E0EC);
  static const Color onSurfaceVariantLight = Color(0xFF49454F);

  // Error
  static const Color errorLight = Color(0xFFBA1A1A);
  static const Color onErrorLight = Color(0xFFFFFFFF);
  static const Color errorContainerLight = Color(0xFFFFDAD6);
  static const Color onErrorContainerLight = Color(0xFF410002);

  // Outline (bordas)
  static const Color outlineLight = Color(0xFF79747E);
  static const Color outlineVariantLight = Color(0xFFCAC4D0);

  // ===== DARK THEME =====

  static const Color primaryDark = Color(0xFFD0BCFF);
  static const Color onPrimaryDark = Color(0xFF381E72);
  static const Color primaryContainerDark = Color(0xFF4F378B);
  static const Color onPrimaryContainerDark = Color(0xFFEADDFF);

  static const Color secondaryDark = Color(0xFFCCC2DC);
  static const Color onSecondaryDark = Color(0xFF332D41);
  static const Color secondaryContainerDark = Color(0xFF4A4458);
  static const Color onSecondaryContainerDark = Color(0xFFE8DEF8);
```

```
  static const Color surfaceDark = Color(0xFF1D1B20);
  static const Color onSurfaceDark = Color(0xFFE6E0E9);
  static const Color surfaceVariantDark = Color(0xFF49454F);
  static const Color onSurfaceVariantDark = Color(0xFFCAC4D0);

  static const Color errorDark = Color(0xFFFFB4AB);
  static const Color onErrorDark = Color(0xFF690005);
  static const Color errorContainerDark = Color(0xFF93000A);
  static const Color onErrorContainerDark = Color(0xFFFFDAD6);

  static const Color outlineDark = Color(0xFF938F99);
  static const Color outlineVariantDark = Color(0xFF49454F);

  // ===== CORES ESPECÍFICAS DE AUTENTICAÇÃO =====

  // Input fields
  static const Color inputFillLight = Color(0xFFF5F5F5);
  static const Color inputFillDark = Color(0xFF2A2A2A);

  // Success
  static const Color successLight = Color(0xFF388E3C);
  static const Color successDark = Color(0xFF81C784);

  // Info
  static const Color infoLight = Color(0xFF1976D2);
  static const Color infoDark = Color(0xFF64B5F6);

  // Social buttons
  static const Color googleRed = Color(0xFFDB4437);
  static const Color githubBlack = Color(0xFF24292E);
  static const Color facebookBlue = Color(0xFF1877F2);
}
```

2.2. Tipografia (Typography)

dart

```dart
// lib/theme/app_typography.dart

import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';

class AppTypography {
  // Fonte base: Roboto (padrão Material) ou Inter (mais moderna)
  static String fontFamily = 'Inter'; // ou 'Roboto'

  // Display (títulos grandes - raramente usado em auth)
  static TextStyle displayLarge = GoogleFonts.getFont(
    fontFamily,
    fontSize: 57,
    fontWeight: FontWeight.w400,
    letterSpacing: -0.25,
  );

  // Headlines (títulos de tela)
  static TextStyle headlineLarge = GoogleFonts.getFont(
    fontFamily,
    fontSize: 32,
    fontWeight: FontWeight.w600,
    letterSpacing: 0,
  );

  static TextStyle headlineMedium = GoogleFonts.getFont(
    fontFamily,
    fontSize: 28,
    fontWeight: FontWeight.w600,
    letterSpacing: 0,
  );

  static TextStyle headlineSmall = GoogleFonts.getFont(
    fontFamily,
    fontSize: 24,
    fontWeight: FontWeight.w600,
    letterSpacing: 0,
  );

  // Títulos (subtítulos de seção)
  static TextStyle titleLarge = GoogleFonts.getFont(
    fontFamily,
    fontSize: 22,
    fontWeight: FontWeight.w500,
    letterSpacing: 0,
  );
```

```
  static TextStyle titleMedium = GoogleFonts.getFont(
    fontFamily,
    fontSize: 16,
    fontWeight: FontWeight.w500,
    letterSpacing: 0.15,
  );

  static TextStyle titleSmall = GoogleFonts.getFont(
    fontFamily,
    fontSize: 14,
    fontWeight: FontWeight.w500,
    letterSpacing: 0.1,
  );

  // Body (texto corrido)
  static TextStyle bodyLarge = GoogleFonts.getFont(
    fontFamily,
    fontSize: 16,
    fontWeight: FontWeight.w400,
    letterSpacing: 0.5,
  );

  static TextStyle bodyMedium = GoogleFonts.getFont(
    fontFamily,
    fontSize: 14,
    fontWeight: FontWeight.w400,
    letterSpacing: 0.25,
  );

  static TextStyle bodySmall = GoogleFonts.getFont(
    fontFamily,
    fontSize: 12,
    fontWeight: FontWeight.w400,
    letterSpacing: 0.4,
  );

  // Labels (labels de campos, botões)
  static TextStyle labelLarge = GoogleFonts.getFont(
    fontFamily,
    fontSize: 14,
    fontWeight: FontWeight.w500,
    letterSpacing: 0.1,
  );

  static TextStyle labelMedium = GoogleFonts.getFont(
    fontFamily,
```

```
    fontSize: 12,
    fontWeight: FontWeight.w500,
    letterSpacing: 0.5,
  );

  static TextStyle labelSmall = GoogleFonts.getFont(
    fontFamily,
    fontSize: 11,
    fontWeight: FontWeight.w500,
    letterSpacing: 0.5,
  );
}
```

2.3. ThemeData Completo

dart

```dart
// lib/theme/app_theme.dart

import 'package:flutter/material.dart';
import 'app_colors.dart';
import 'app_typography.dart';

class AppTheme {
  static ThemeData lightTheme = ThemeData(
    useMaterial3: true,
    brightness: Brightness.light,

    // Color Scheme
    colorScheme: ColorScheme.light(
      primary: AppColors.primaryLight,
      onPrimary: AppColors.onPrimaryLight,
      primaryContainer: AppColors.primaryContainerLight,
      onPrimaryContainer: AppColors.onPrimaryContainerLight,

      secondary: AppColors.secondaryLight,
      onSecondary: AppColors.onSecondaryLight,
      secondaryContainer: AppColors.secondaryContainerLight,
      onSecondaryContainer: AppColors.onSecondaryContainerLight,

      surface: AppColors.surfaceLight,
      onSurface: AppColors.onSurfaceLight,
      surfaceVariant: AppColors.surfaceVariantLight,
      onSurfaceVariant: AppColors.onSurfaceVariantLight,

      error: AppColors.errorLight,
      onError: AppColors.onErrorLight,
      errorContainer: AppColors.errorContainerLight,
      onErrorContainer: AppColors.onErrorContainerLight,

      outline: AppColors.outlineLight,
      outlineVariant: AppColors.outlineVariantLight,
    ),

    // Typography
    textTheme: TextTheme(
      displayLarge: AppTypography.displayLarge,
      headlineLarge: AppTypography.headlineLarge,
      headlineMedium: AppTypography.headlineMedium,
      headlineSmall: AppTypography.headlineSmall,
      titleLarge: AppTypography.titleLarge,
      titleMedium: AppTypography.titleMedium,
      titleSmall: AppTypography.titleSmall,
```

```
    bodyLarge: AppTypography.bodyLarge,
    bodyMedium: AppTypography.bodyMedium,
    bodySmall: AppTypography.bodySmall,
    labelLarge: AppTypography.labelLarge,
    labelMedium: AppTypography.labelMedium,
    labelSmall: AppTypography.labelSmall,
  ),

  // Input Decoration Theme
  inputDecorationTheme: InputDecorationTheme(
    filled: true,
    fillColor: AppColors.inputFillLight,

    // Bordas
    border: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide.none,
    ),
    enabledBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide.none,
    ),
    focusedBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide(
        color: AppColors.primaryLight,
        width: 2,
      ),
    ),
    errorBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide(
        color: AppColors.errorLight,
        width: 1,
      ),
    ),
    focusedErrorBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide(
        color: AppColors.errorLight,
        width: 2,
      ),
    ),

    // Padding interno
    contentPadding: const EdgeInsets.symmetric(
      horizontal: 16,
```

```
      vertical: 16,
    ),

    // Estilo de labels e hints
    labelStyle: AppTypography.bodyMedium,
    hintStyle: AppTypography.bodyMedium.copyWith(
      color: AppColors.onSurfaceVariantLight.withOpacity(0.6),
    ),
    errorStyle: AppTypography.bodySmall.copyWith(
      color: AppColors.errorLight,
    ),
  ),

  // Elevated Button Theme
  elevatedButtonTheme: ElevatedButtonThemeData(
    style: ElevatedButton.styleFrom(
      backgroundColor: AppColors.primaryLight,
      foregroundColor: AppColors.onPrimaryLight,
      elevation: 2,
      shadowColor: Colors.black26,

      // Shape
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(12),
      ),

      // Padding
      padding: const EdgeInsets.symmetric(
        horizontal: 24,
        vertical: 16,
      ),

      // Text style
      textStyle: AppTypography.labelLarge,

      // Tamanho mínimo
      minimumSize: const Size(0, 48),
    ),
  ),

  // Outlined Button Theme
  outlinedButtonTheme: OutlinedButtonThemeData(
    style: OutlinedButton.styleFrom(
      foregroundColor: AppColors.primaryLight,

      // Borda
      side: BorderSide(
```

```
      color: AppColors.outlineLight,
      width: 1,
    ),

    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(12),
    ),

    padding: const EdgeInsets.symmetric(
      horizontal: 24,
      vertical: 16,
    ),

    textStyle: AppTypography.labelLarge,
    minimumSize: const Size(0, 48),
  ),
),

// Text Button Theme
textButtonTheme: TextButtonThemeData(
  style: TextButton.styleFrom(
    foregroundColor: AppColors.primaryLight,
    textStyle: AppTypography.labelLarge,
    padding: const EdgeInsets.symmetric(
      horizontal: 16,
      vertical: 12,
    ),
  ),
),

// App Bar Theme
appBarTheme: AppBarTheme(
  backgroundColor: AppColors.surfaceLight,
  foregroundColor: AppColors.onSurfaceLight,
  elevation: 0,
  centerTitle: true,
  titleTextStyle: AppTypography.titleLarge.copyWith(
    color: AppColors.onSurfaceLight,
  ),
),

// Card Theme
cardTheme: CardTheme(
  color: AppColors.surfaceLight,
  elevation: 2,
  shadowColor: Colors.black12,
  shape: RoundedRectangleBorder(
```

```
      borderRadius: BorderRadius.circular(16),
    ),
    margin: const EdgeInsets.all(8),
  ),

  // Divider Theme
  dividerTheme: DividerThemeData(
    color: AppColors.outlineVariantLight,
    thickness: 1,
    space: 1,
  ),
);

// ===== DARK THEME =====

static ThemeData darkTheme = ThemeData(
  useMaterial3: true,
  brightness: Brightness.dark,

  colorScheme: ColorScheme.dark(
    primary: AppColors.primaryDark,
    onPrimary: AppColors.onPrimaryDark,
    primaryContainer: AppColors.primaryContainerDark,
    onPrimaryContainer: AppColors.onPrimaryContainerDark,

    secondary: AppColors.secondaryDark,
    onSecondary: AppColors.onSecondaryDark,
    secondaryContainer: AppColors.secondaryContainerDark,
    onSecondaryContainer: AppColors.onSecondaryContainerDark,

    surface: AppColors.surfaceDark,
    onSurface: AppColors.onSurfaceDark,
    surfaceVariant: AppColors.surfaceVariantDark,
    onSurfaceVariant: AppColors.onSurfaceVariantDark,

    error: AppColors.errorDark,
    onError: AppColors.onErrorDark,
    errorContainer: AppColors.errorContainerDark,
    onErrorContainer: AppColors.onErrorContainerDark,

    outline: AppColors.outlineDark,
    outlineVariant: AppColors.outlineVariantDark,
  ),

  // Reutilizar textTheme (funciona em ambos os temas)
  textTheme: TextTheme(
    displayLarge: AppTypography.displayLarge,
```

```
    headlineLarge: AppTypography.headlineLarge,
    headlineMedium: AppTypography.headlineMedium,
    headlineSmall: AppTypography.headlineSmall,
    titleLarge: AppTypography.titleLarge,
    titleMedium: AppTypography.titleMedium,
    titleSmall: AppTypography.titleSmall,
    bodyLarge: AppTypography.bodyLarge,
    bodyMedium: AppTypography.bodyMedium,
    bodySmall: AppTypography.bodySmall,
    labelLarge: AppTypography.labelLarge,
    labelMedium: AppTypography.labelMedium,
    labelSmall: AppTypography.labelSmall,
  ),

  // Input Decoration para dark theme
  inputDecorationTheme: InputDecorationTheme(
    filled: true,
    fillColor: AppColors.inputFillDark,

    border: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide.none,
    ),
    enabledBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide.none,
    ),
    focusedBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide(
        color: AppColors.primaryDark,
        width: 2,
      ),
    ),
    errorBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide(
        color: AppColors.errorDark,
        width: 1,
      ),
    ),
    focusedErrorBorder: OutlineInputBorder(
      borderRadius: BorderRadius.circular(12),
      borderSide: BorderSide(
        color: AppColors.errorDark,
        width: 2,
      ),
```

```
    ),

    contentPadding: const EdgeInsets.symmetric(
      horizontal: 16,
      vertical: 16,
    ),

    labelStyle: AppTypography.bodyMedium,
    hintStyle: AppTypography.bodyMedium.copyWith(
      color: AppColors.onSurfaceVariantDark.withOpacity(0.6),
    ),
    errorStyle: AppTypography.bodySmall.copyWith(
      color: AppColors.errorDark,
    ),
  ),

  // Botões para dark theme
  elevatedButtonTheme: ElevatedButtonThemeData(
    style: ElevatedButton.styleFrom(
      backgroundColor: AppColors.primaryDark,
      foregroundColor: AppColors.onPrimaryDark,
      elevation: 2,
      shadowColor: Colors.black54,
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(12),
      ),
      padding: const EdgeInsets.symmetric(
        horizontal: 24,
        vertical: 16,
      ),
      textStyle: AppTypography.labelLarge,
      minimumSize: const Size(0, 48),
    ),
  ),

  outlinedButtonTheme: OutlinedButtonThemeData(
    style: OutlinedButton.styleFrom(
      foregroundColor: AppColors.primaryDark,
      side: BorderSide(
        color: AppColors.outlineDark,
        width: 1,
      ),
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(12),
      ),
      padding: const EdgeInsets.symmetric(
        horizontal: 24,
```

```
        vertical: 16,
      ),
      textStyle: AppTypography.labelLarge,
      minimumSize: const Size(0, 48),
    ),
  ),

  textButtonTheme: TextButtonThemeData(
    style: TextButton.styleFrom(
      foregroundColor: AppColors.primaryDark,
      textStyle: AppTypography.labelLarge,
      padding: const EdgeInsets.symmetric(
        horizontal: 16,
        vertical: 12,
      ),
    ),
  ),

  appBarTheme: AppBarTheme(
    backgroundColor: AppColors.surfaceDark,
    foregroundColor: AppColors.onSurfaceDark,
    elevation: 0,
    centerTitle: true,
    titleTextStyle: AppTypography.titleLarge.copyWith(
      color: AppColors.onSurfaceDark,
    ),
  ),

  cardTheme: CardTheme(
    color: AppColors.surfaceDark,
    elevation: 2,
    shadowColor: Colors.black54,
    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(16),
    ),
    margin: const EdgeInsets.all(8),
  ),

  dividerTheme: DividerThemeData(
    color: AppColors.outlineVariantDark,
    thickness: 1,
    space: 1,
  ),
 );
}
```

Parte 3: Componentes Reutilizáveis

3.1. CustomTextField

dart

```dart
// lib/widgets/custom_text_field.dart

import 'package:flutter/material.dart';
import 'package:flutter/services.dart';

/// Campo de texto personalizado para formulários de autenticação
///
/// Suporta validação, ícones, sufixo, e formatação
class CustomTextField extends StatelessWidget {
  final TextEditingController controller;
  final String label;
  final String? hint;
  final String? Function(String?)? validator;
  final TextInputType keyboardType;
  final bool obscureText;
  final Widget? prefixIcon;
  final Widget? suffixIcon;
  final bool enabled;
  final int? maxLines;
  final int? maxLength;
  final List<TextInputFormatter>? inputFormatters;
  final void Function(String)? onChanged;
  final FocusNode? focusNode;
  final TextInputAction? textInputAction;
  final void Function(String)? onFieldSubmitted;

  const CustomTextField({
    super.key,
    required this.controller,
    required this.label,
    this.hint,
    this.validator,
    this.keyboardType = TextInputType.text,
    this.obscureText = false,
    this.prefixIcon,
    this.suffixIcon,
    this.enabled = true,
    this.maxLines = 1,
    this.maxLength,
    this.inputFormatters,
    this.onChanged,
    this.focusNode,
    this.textInputAction,
    this.onFieldSubmitted,
  });
```

```dart
  @override
  Widget build(BuildContext context) {
    return TextFormField(
      controller: controller,
      validator: validator,
      keyboardType: keyboardType,
      obscureText: obscureText,
      enabled: enabled,
      maxLines: obscureText ? 1 : maxLines,
      maxLength: maxLength,
      inputFormatters: inputFormatters,
      onChanged: onChanged,
      focusNode: focusNode,
      textInputAction: textInputAction,
      onFieldSubmitted: onFieldSubmitted,

      decoration: InputDecoration(
        labelText: label,
        hintText: hint,
        prefixIcon: prefixIcon,
        suffixIcon: suffixIcon,

        // Contador de caracteres estilizado
        counterStyle: Theme.of(context).textTheme.bodySmall?.copyWith(
          color: Theme.of(context).colorScheme.onSurfaceVariant,
        ),
      ),
    );
  }
}

/// Campo de email pré-configurado
class EmailTextField extends StatelessWidget {
  final TextEditingController controller;
  final String? Function(String?)? validator;
  final void Function(String)? onChanged;
  final TextInputAction? textInputAction;
  final void Function(String)? onFieldSubmitted;

  const EmailTextField({
    super.key,
    required this.controller,
    this.validator,
    this.onChanged,
    this.textInputAction,
    this.onFieldSubmitted,
  });
```

```dart
  @override
  Widget build(BuildContext context) {
    return CustomTextField(
      controller: controller,
      label: 'Email',
      hint: 'seu@email.com',
      keyboardType: TextInputType.emailAddress,
      prefixIcon: const Icon(Icons.email_outlined),
      validator: validator,
      onChanged: onChanged,
      textInputAction: textInputAction,
      onFieldSubmitted: onFieldSubmitted,
    );
  }
}

/// Campo de senha com toggle de visibilidade
class PasswordTextField extends StatefulWidget {
  final TextEditingController controller;
  final String label;
  final String? hint;
  final String? Function(String?)? validator;
  final void Function(String)? onChanged;
  final TextInputAction? textInputAction;
  final void Function(String)? onFieldSubmitted;

  const PasswordTextField({
    super.key,
    required this.controller,
    this.label = 'Senha',
    this.hint,
    this.validator,
    this.onChanged,
    this.textInputAction,
    this.onFieldSubmitted,
  });

  @override
  State<PasswordTextField> createState() => _PasswordTextFieldState();
}

class _PasswordTextFieldState extends State<PasswordTextField> {
  bool _obscureText = true;

  @override
  Widget build(BuildContext context) {
```

```
    return CustomTextField(
      controller: widget.controller,
      label: widget.label,
      hint: widget.hint ?? '••••••••',
      obscureText: _obscureText,
      prefixIcon: const Icon(Icons.lock_outline),
      suffixIcon: IconButton(
        icon: Icon(
          _obscureText ? Icons.visibility : Icons.visibility_off,
        ),
        onPressed: () {
          setState(() => _obscureText = !_obscureText);
        },
      ),
      validator: widget.validator,
      onChanged: widget.onChanged,
      textInputAction: widget.textInputAction,
      onFieldSubmitted: widget.onFieldSubmitted,
    );
  }
}
```

3.2. AuthButton (Botão de Autenticação)

dart

```dart
// lib/widgets/auth_button.dart

import 'package:flutter/material.dart';

/// Botão primário para ações de autenticação
///
/// Suporta estado de loading e desabilitado
class AuthButton extends StatelessWidget {
  final String text;
  final VoidCallback? onPressed;
  final bool isLoading;
  final bool isFullWidth;
  final Widget? icon;
  final Color? backgroundColor;
  final Color? foregroundColor;

  const AuthButton({
    super.key,
    required this.text,
    this.onPressed,
    this.isLoading = false,
    this.isFullWidth = true,
    this.icon,
    this.backgroundColor,
    this.foregroundColor,
  });

  @override
  Widget build(BuildContext context) {
    final button = ElevatedButton(
      onPressed: isLoading ? null : onPressed,
      style: ElevatedButton.styleFrom(
        backgroundColor: backgroundColor,
        foregroundColor: foregroundColor,
      ),
      child: isLoading
          ? SizedBox(
              height: 20,
              width: 20,
              child: CircularProgressIndicator(
                strokeWidth: 2,
                color: foregroundColor ??
                    Theme.of(context).colorScheme.onPrimary,
              ),
            )
          : icon != null
```

```
            ? Row(
                mainAxisSize: MainAxisSize.min,
                children: [
                  icon!,
                  const SizedBox(width: 8),
                  Text(text),
                ],
              )
            : Text(text),
    );

    return isFullWidth
        ? SizedBox(
            width: double.infinity,
            height: 48,
            child: button,
          )
        : button;
  }
}

/// Botão secundário (outline)
class AuthOutlineButton extends StatelessWidget {
  final String text;
  final VoidCallback? onPressed;
  final bool isFullWidth;
  final Widget? icon;

  const AuthOutlineButton({
    super.key,
    required this.text,
    this.onPressed,
    this.isFullWidth = true,
    this.icon,
  });

  @override
  Widget build(BuildContext context) {
    final button = OutlinedButton(
      onPressed: onPressed,
      child: icon != null
          ? Row(
              mainAxisSize: MainAxisSize.min,
              children: [
                icon!,
                const SizedBox(width: 8),
                Text(text),
```

```
            ],
          )
        : Text(text),
    );

    return isFullWidth
        ? SizedBox(
            width: double.infinity,
            height: 48,
            child: button,
          )
        : button;
  }
}

/// Botão de login social
class SocialButton extends StatelessWidget {
  final String provider; // 'google', 'github', 'facebook', etc
  final VoidCallback onPressed;
  final bool isFullWidth;

  const SocialButton({
    super.key,
    required this.provider,
    required this.onPressed,
    this.isFullWidth = true,
  });

  @override
  Widget build(BuildContext context) {
    final config = _getSocialConfig(provider);

    final button = OutlinedButton(
      onPressed: onPressed,
      style: OutlinedButton.styleFrom(
        side: BorderSide(
          color: Theme.of(context).colorScheme.outlineVariant,
          width: 1,
        ),
      ),
      child: Row(
        mainAxisSize: MainAxisSize.min,
        children: [
          config.icon,
          const SizedBox(width: 12),
          Text(
            'Continuar com ${config.displayName}',
```

```
            style: TextStyle(
              color: Theme.of(context).colorScheme.onSurface,
            ),
          ),
        ],
      ),
    );

    return isFullWidth
        ? SizedBox(
            width: double.infinity,
            height: 48,
            child: button,
          )
        : button;
  }

  _SocialConfig _getSocialConfig(String provider) {
    switch (provider.toLowerCase()) {
      case 'google':
        return _SocialConfig(
          displayName: 'Google',
          icon: Image.asset(
            'assets/icons/google_logo.png',
            height: 24,
            width: 24,
          ),
        );
      case 'github':
        return _SocialConfig(
          displayName: 'GitHub',
          icon: const Icon(Icons.code, size: 24),
        );
      case 'facebook':
        return _SocialConfig(
          displayName: 'Facebook',
          icon: const Icon(Icons.facebook, size: 24),
        );
      default:
        return _SocialConfig(
          displayName: provider,
          icon: const Icon(Icons.login, size: 24),
        );
    }
  }
}
```

```
class _SocialConfig {
  final String displayName;
  final Widget icon;

  _SocialConfig({
    required this.displayName,
    required this.icon,
  });
}
```

3.3. AuthHeader (Cabeçalho de Tela)

dart

```dart
// lib/widgets/auth_header.dart

import 'package:flutter/material.dart';

/// Cabeçalho padronizado para telas de autenticação
///
/// Inclui logo, título e subtítulo
class AuthHeader extends StatelessWidget {
  final String? logoPath;
  final IconData? logoIcon;
  final String title;
  final String? subtitle;

  const AuthHeader({
    super.key,
    this.logoPath,
    this.logoIcon,
    required this.title,
    this.subtitle,
  });

  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        // Logo ou ícone
        if (logoPath != null)
          Image.asset(
            logoPath!,
            height: 64,
            width: 64,
          )
        else if (logoIcon != null)
          Container(
            height: 64,
            width: 64,
            decoration: BoxDecoration(
              color: Theme.of(context).colorScheme.primaryContainer,
              borderRadius: BorderRadius.circular(16),
            ),
            child: Icon(
              logoIcon,
              size: 32,
              color: Theme.of(context).colorScheme.onPrimaryContainer,
            ),
          ),
```

```
        if (logoPath != null || logoIcon != null)
          const SizedBox(height: 24),

        // Título
        Text(
          title,
          style: Theme.of(context).textTheme.headlineSmall,
          textAlign: TextAlign.center,
        ),

        // Subtítulo (opcional)
        if (subtitle != null) ...[
          const SizedBox(height: 8),
          Text(
            subtitle!,
            style: Theme.of(context).textTheme.bodyMedium?.copyWith(
              color: Theme.of(context).colorScheme.onSurfaceVariant,
            ),
            textAlign: TextAlign.center,
          ),
        ],
      ],
    );
  }
}
```

3.4. AuthDivider (Divisor "OU")

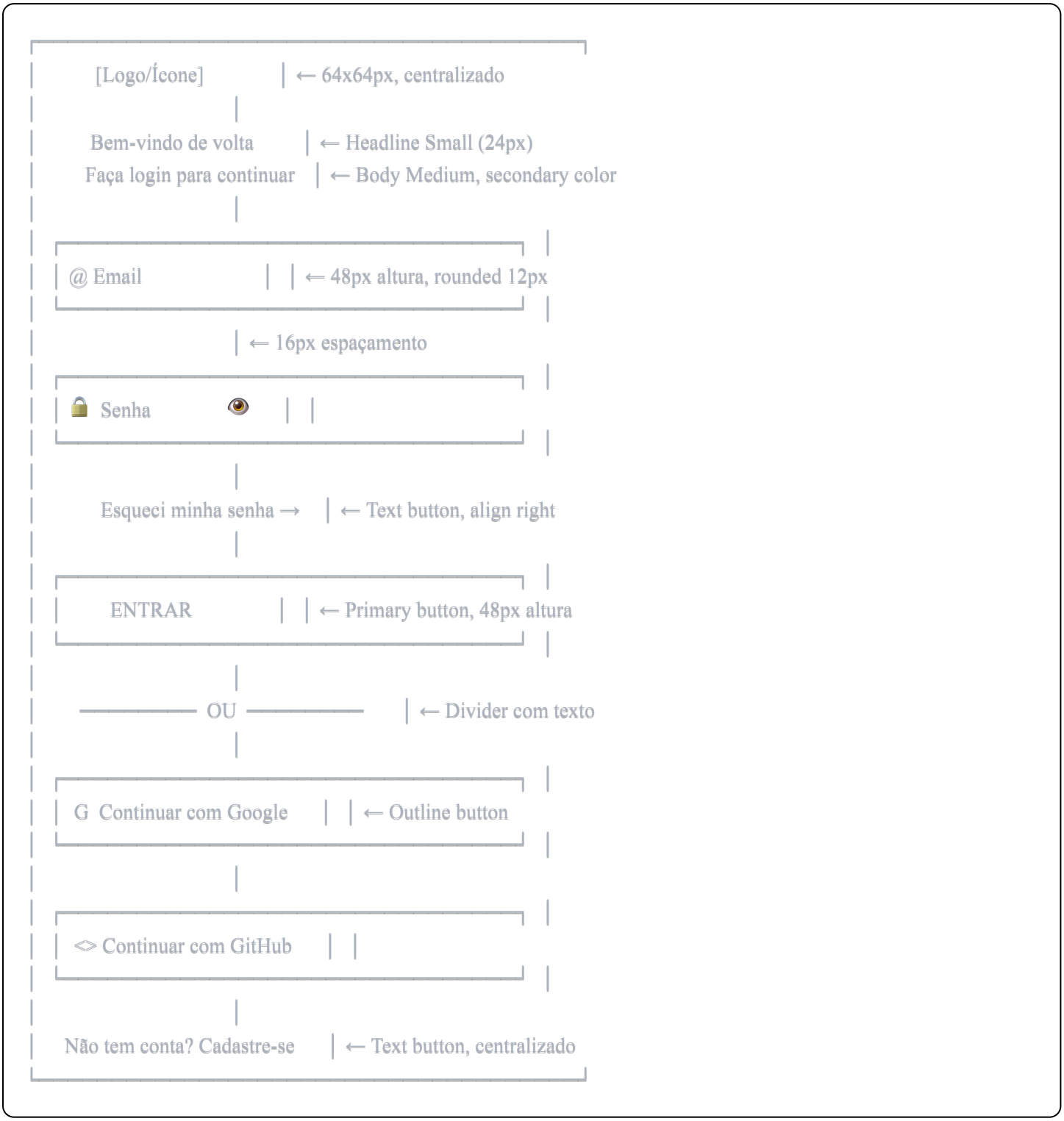
dart

```dart
// lib/widgets/auth_divider.dart

import 'package:flutter/material.dart';

/// Divisor com texto "OU" para separar métodos de autenticação
class AuthDivider extends StatelessWidget {
  final String text;

  const AuthDivider({
    super.key,
    this.text = 'OU',
  });

  @override
  Widget build(BuildContext context) {
    return Row(
      children: [
        Expanded(
          child: Divider(
            color: Theme.of(context).colorScheme.outlineVariant,
          ),
        ),
        Padding(
          padding: const EdgeInsets.symmetric(horizontal: 16),
          child: Text(
            text,
            style: Theme.of(context).textTheme.bodySmall?.copyWith(
              color: Theme.of(context).colorScheme.onSurfaceVariant,
            ),
          ),
        ),
        Expanded(
          child: Divider(
            color: Theme.of(context).colorScheme.outlineVariant,
          ),
        ),
      ],
    );
  }
}
```

Parte 4: Especificações Detalhadas por Tela

4.1. Tela de Login

Anatomia:



Espaçamentos:

- Padding externo: 24px (laterais e topo/fundo)
- Logo ao título: 24px
- Título ao subtítulo: 8px

- Subtítulo aos campos: 32px
- Entre campos: 16px
- Campo ao "esqueci senha": 8px
- Esqueci senha ao botão: 24px
- Botão ao divisor: 24px
- Divisor aos sociais: 24px
- Entre botões sociais: 12px
- Sociais ao "cadastre-se": 24px

Cores (Light Theme):

- Background: Surface (#FEF7FF)
- Título: OnSurface (#1D1B20)
- Subtítulo: OnSurfaceVariant (#49454F)
- Campos: InputFill (#F5F5F5) com outline quando focado
- Botão primário: Primary (#6750A4)
- Botões sociais: Outline (#79747E)

4.2. Tela de Cadastro

Anatomia:

Indicador de força de senha:

dart

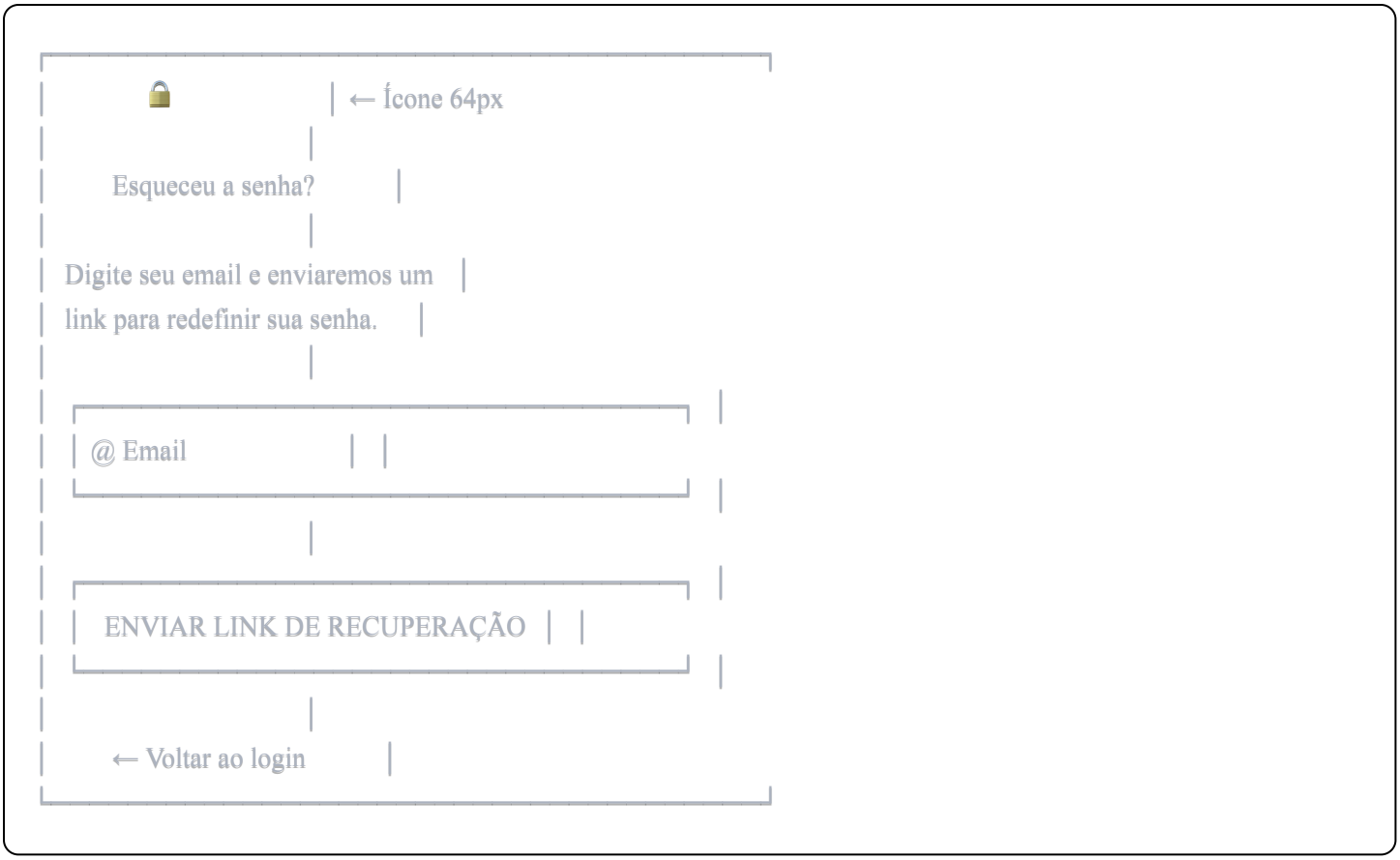
```dart
// lib/widgets/password_strength_indicator.dart

import 'package:flutter/material.dart';

class PasswordStrengthIndicator extends StatelessWidget {
  final String password;

  const PasswordStrengthIndicator({
    super.key,
    required this.password,
  });

  @override
  Widget build(BuildContext context) {
    final strength = _calculateStrength(password);

    return Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Row(
          children: List.generate(4, (index) {
            return Expanded(
              child: Container(
                height: 4,
                margin: EdgeInsets.only(
                  right: index < 3 ? 4 : 0,
                ),
                decoration: BoxDecoration(
                  color: index < strength.bars
                      ? strength.color
                      : Theme.of(context).colorScheme.surfaceVariant,
                  borderRadius: BorderRadius.circular(2),
                ),
              ),
            );
          }),
        ),
        if (password.isNotEmpty) ...[
          const SizedBox(height: 4),
          Text(
            'Força: ${strength.label}',
            style: Theme.of(context).textTheme.bodySmall?.copyWith(
              color: strength.color,
            ),
          ),
        ],
```

```
    ],
  );
}

_PasswordStrength _calculateStrength(String password) {
  if (password.isEmpty) {
    return _PasswordStrength(
      bars: 0,
      label: '',
      color: Colors.transparent,
    );
  }

  int score = 0;

  // Comprimento
  if (password.length >= 8) score++;
  if (password.length >= 12) score++;

  // Tem letra minúscula
  if (password.contains(RegExp(r'[a-z]'))) score++;

  // Tem letra maiúscula
  if (password.contains(RegExp(r'[A-Z]'))) score++;

  // Tem número
  if (password.contains(RegExp(r'[0-9]'))) score++;

  // Tem caractere especial
  if (password.contains(RegExp(r'[!@#$%^&*(),.?":{}|<>]'))) score++;

  // Mapear score para strength
  if (score <= 2) {
    return _PasswordStrength(
      bars: 1,
      label: 'Fraca',
      color: Colors.red,
    );
  } else if (score <= 4) {
    return _PasswordStrength(
      bars: 2,
      label: 'Média',
      color: Colors.orange,
    );
  } else if (score == 5) {
    return _PasswordStrength(
      bars: 3,
```

```
        label: 'Forte',
        color: Colors.green,
      );
    } else {
      return _PasswordStrength(
        bars: 4,
        label: 'Muito Forte',
        color: Colors.green[700]!,
      );
    }
  }
}

class _PasswordStrength {
  final int bars;
  final String label;
  final Color color;

  _PasswordStrength({
    required this.bars,
    required this.label,
    required this.color,
  });
}
```

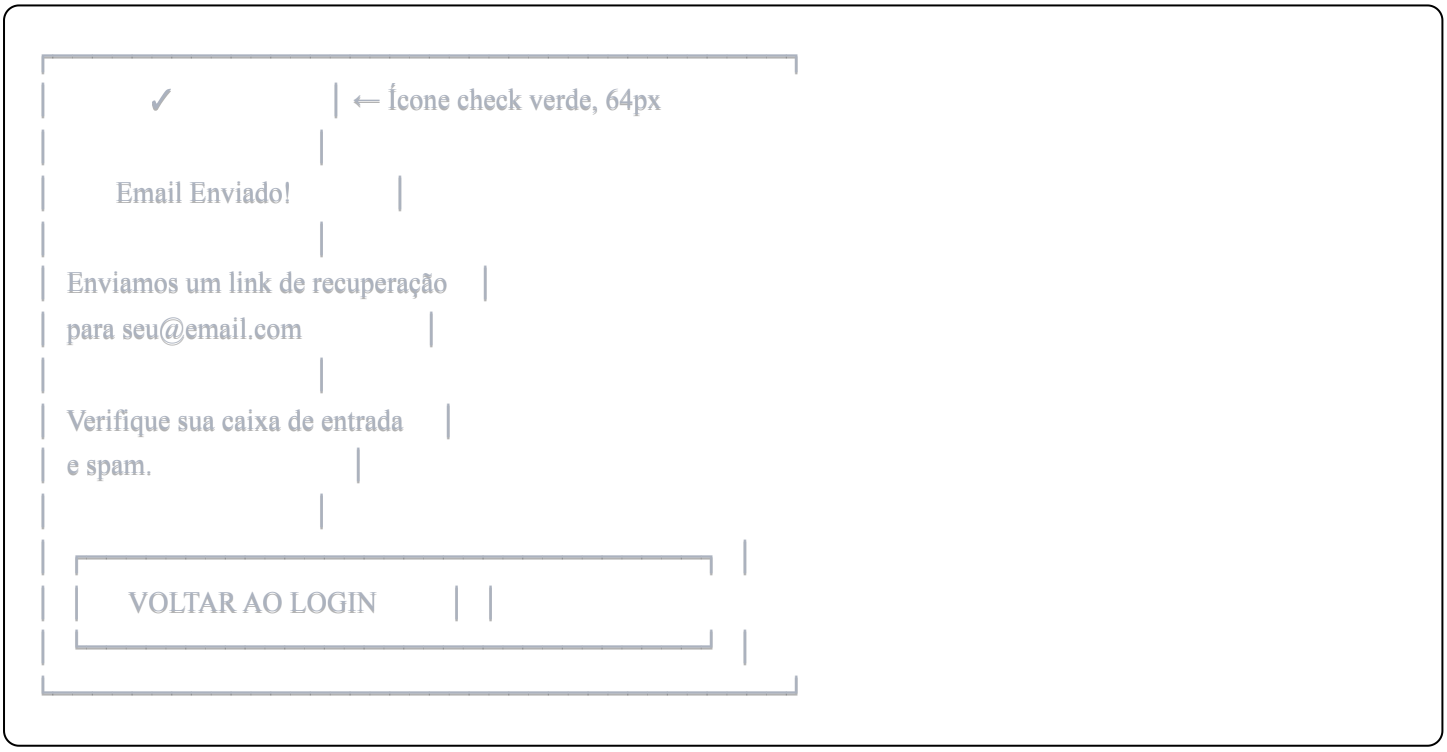
4.3. Tela de Recuperação de Senha

Anatomia:



4.4. Tela de Sucesso (após envio de email)

Anatomia:



Parte 5: Prompts para Geração de Protótipos com IA

5.1. Prompt para Tela de Login

Crie um protótipo de tela de login para aplicativo mobile usando Material Design 3.

ESPECIFICAÇÕES:

Layout:
- Orientação: Vertical (portrait)
- Resolução: 375x812px (iPhone X)
- Padding: 24px nas laterais
- Background: Branco suave (#FAFAFA)

Elementos (de cima para baixo):

1. Logo/Ícone:
   - Posição: Centralizado, 60px do topo
   - Tamanho: 64x64px
   - Estilo: Ícone de cadeado em círculo roxo (#6750A4)
   - Background do círculo: Roxo claro (#EADDFF)

2. Título:
   - Texto: "Bem-vindo de volta"
   - Font: Inter Bold, 24px
   - Cor: #1D1B20
   - Alinhamento: Centro
   - Margem superior: 24px do logo

3. Subtítulo:
   - Texto: "Faça login para continuar"
   - Font: Inter Regular, 14px
   - Cor: #49454F
   - Alinhamento: Centro
   - Margem superior: 8px do título

4. Campo Email:
   - Margem superior: 32px do subtítulo
   - Altura: 56px
   - Background: #F5F5F5
   - Border radius: 12px
   - Label: "Email" (cinza #49454F)
   - Ícone prefixo: @ (cinza)
   - Placeholder: "seu@email.com"

5. Campo Senha:

- Margem superior: 16px do campo email
- Mesmas specs do campo email
- Label: "Senha"
- Ícone prefixo: Cadeado
- Ícone sufixo: Olho (toggle visibilidade)
- Placeholder: "••••••••"

6. Link "Esqueci senha":
- Margem superior: 8px do campo senha
- Alinhamento: Direita
- Font: Inter Medium, 14px
- Cor: #6750A4
- Texto: "Esqueci minha senha"

7. Botão Login:
- Margem superior: 24px do link
- Largura: 100%
- Altura: 48px
- Background: #6750A4 (roxo primário)
- Texto: "ENTRAR" (branco, Inter SemiBold, 14px)
- Border radius: 12px
- Sombra sutil

8. Divisor:
- Margem superior: 24px do botão
- Linha cinza clara com texto "OU" no centro
- Font: Inter Regular, 12px, cor #49454F

9. Botão Google:
- Margem superior: 24px do divisor
- Largura: 100%
- Altura: 48px
- Background: Branco
- Border: 1px sólido #E0E0E0
- Ícone: Logo Google (24px)
- Texto: "Continuar com Google" (Inter Medium, 14px, #1D1B20)
- Border radius: 12px

10. Botão GitHub:
- Margem superior: 12px do botão Google
- Mesmas specs do botão Google
- Ícone: GitHub (preto)
- Texto: "Continuar com GitHub"

11. Link Cadastro:
- Margem superior: 24px do último botão
- Alinhamento: Centro

- Font: Inter Regular, 14px
- Texto: "Não tem conta? " (cinza) + "Cadastre-se" (roxo #6750A4, bold)

ESTILO:
- Design limpo, moderno, minimalista
- Seguir princípios Material Design 3
- Espaçamento generoso entre elementos
- Foco em usabilidade e clareza

5.2. Prompt para Tela de Cadastro

Crie um protótipo de tela de cadastro para aplicativo mobile usando Material Design 3.

ESPECIFICAÇÕES:

Layout:
- Orientação: Vertical (portrait), scrollável
- Resolução: 375x812px (iPhone X)
- Padding: 24px nas laterais
- Background: Branco suave (#FAFAFA)

Elementos (de cima para baixo):

1. Logo/Ícone:
   - Posição: Centralizado, 40px do topo
   - Tamanho: 64x64px
   - Estilo: Ícone de usuário em círculo roxo (#6750A4)
   - Background do círculo: Roxo claro (#EADDFF)

2. Título:
   - Texto: "Criar Conta"
   - Font: Inter Bold, 24px
   - Cor: #1D1B20
   - Alinhamento: Centro
   - Margem superior: 24px do logo

3. Subtítulo:
   - Texto: "Junte-se a nós em poucos passos"
   - Font: Inter Regular, 14px
   - Cor: #49454F
   - Alinhamento: Centro
   - Margem superior: 8px do título

4. Campo Nome:
   - Margem superior: 32px do subtítulo
   - Altura: 56px

- Background: #F5F5F5
- Border radius: 12px
- Label: "Nome completo"
- Ícone prefixo: Pessoa
- Placeholder: "João Silva"

5. Campo Email:
- Margem superior: 16px do campo nome
- Mesmas specs do campo nome
- Label: "Email"
- Ícone: @
- Placeholder: "seu@email.com"

6. Campo Senha:
- Margem superior: 16px do campo email
- Mesmas specs
- Label: "Senha"
- Ícone prefixo: Cadeado
- Ícone sufixo: Olho
- Placeholder: "••••••••"

7. Indicador de Força:
- Margem superior: 8px do campo senha
- 4 barras horizontais (3px altura cada)
- Cores: Vermelho (fraca), Laranja (média), Verde (forte)
- Label: "Força: Média" (cor correspondente)

8. Campo Confirmar Senha:
- Margem superior: 16px do indicador
- Mesmas specs do campo senha
- Label: "Confirmar senha"

9. Checkbox Termos:
- Margem superior: 20px
- Checkbox: 20x20px, roxo quando marcado
- Texto: "Aceito os Termos e Condições"
- Font: Inter Regular, 13px
- "Termos" deve ser link roxo sublinhado

10. Botão Criar Conta:
- Margem superior: 24px do checkbox
- Largura: 100%
- Altura: 48px
- Background: #6750A4
- Texto: "CRIAR CONTA" (branco, Inter SemiBold, 14px)
- Border radius: 12px
- Sombra sutil

11. Divisor "OU":
- Margem superior: 24px
- Linha cinza com texto "OU"

12. Botão Google:
- Margem superior: 24px
- Largura: 100%
- Altura: 48px
- Background: Branco
- Border: 1px #E0E0E0
- Ícone: Google
- Texto: "Continuar com Google"

13. Link Login:
- Margem superior: 24px
- Margem inferior: 24px (bottom padding)
- Alinhamento: Centro
- Texto: "Já tem conta? " + "Faça login" (roxo bold)

ESTILO:
- Design limpo, organizado, scrollável
- Material Design 3
- Campos bem espaçados para facilitar preenchimento
- Foco em clareza e simplicidade

5.3. Prompt para Tela de Recuperação de Senha

Crie um protótipo de tela de recuperação de senha para aplicativo mobile usando Material Design 3.

ESPECIFICAÇÕES:

Layout:
- Orientação: Vertical (portrait)
- Resolução: 375x812px
- Padding: 24px nas laterais
- Background: Branco suave (#FAFAFA)

Elementos (centralizados verticalmente):

1. Ícone:
- Posição: Centralizado
- Tamanho: 80x80px
- Ícone: Cadeado aberto com círculo
- Cor: Roxo (#6750A4)
- Background: Círculo roxo claro (#EADDFF)

2. Título:
- Texto: "Esqueceu a senha?"
- Font: Inter Bold, 24px
- Cor: #1D1B20
- Alinhamento: Centro
- Margem superior: 24px do ícone

3. Descrição:
- Texto: "Digite seu email e enviaremos um link para redefinir sua senha."
- Font: Inter Regular, 14px
- Cor: #49454F
- Alinhamento: Centro
- Margem superior: 16px do título
- Max width: 280px

4. Campo Email:
- Margem superior: 32px da descrição
- Largura: 100%
- Altura: 56px
- Background: #F5F5F5
- Border radius: 12px
- Label: "Email"
- Ícone: @
- Placeholder: "seu@email.com"

5. Botão Enviar:
- Margem superior: 24px do campo
- Largura: 100%
- Altura: 48px
- Background: #6750A4
- Texto: "ENVIAR LINK DE RECUPERAÇÃO"
- Cor texto: Branco
- Font: Inter SemiBold, 14px
- Border radius: 12px

6. Link Voltar:
- Margem superior: 16px do botão
- Alinhamento: Centro
- Ícone: Seta esquerda
- Texto: "Voltar ao login"
- Cor: #6750A4
- Font: Inter Medium, 14px

ESTILO:
- Design focado, sem distrações
- Centralizado verticalmente

- Ícone expressivo para transmitir ação
- Texto claro explicando o processo

5.4. Prompt para Tela de Sucesso

Crie um protótipo de tela de sucesso (email enviado) para aplicativo mobile usando Material Design 3.

ESPECIFICAÇÕES:

Layout:
- Orientação: Vertical (portrait)
- Resolução: 375x812px
- Padding: 24px nas laterais
- Background: Branco suave (#FAFAFA)

Elementos (centralizados verticalmente):

1. Ícone Sucesso:
   - Posição: Centralizado
   - Tamanho: 80x80px
   - Ícone: Check mark em círculo
   - Cor: Verde (#4CAF50)
   - Background: Círculo verde claro (#C8E6C9)
   - Animação sugerida: Fade in + escala

2. Título:
   - Texto: "Email Enviado!"
   - Font: Inter Bold, 28px
   - Cor: #1D1B20
   - Alinhamento: Centro
   - Margem superior: 24px do ícone

3. Mensagem Principal:
   - Texto: "Enviamos um link de recuperação para"
   - Font: Inter Regular, 14px
   - Cor: #49454F
   - Alinhamento: Centro
   - Margem superior: 16px do título

4. Email Destacado:
   - Texto: "seu@email.com"
   - Font: Inter SemiBold, 16px
   - Cor: #6750A4 (roxo)
   - Alinhamento: Centro
   - Margem superior: 4px

5. Instrução:
   - Texto: "Verifique sua caixa de entrada e spam."
   - Font: Inter Regular, 13px
   - Cor: #49454F
   - Alinhamento: Centro
   - Margem superior: 16px

6. Nota Temporal:
   - Texto: "O link expira em 1 hora."
   - Font: Inter Regular, 12px
   - Cor: #F57C00 (laranja)
   - Alinhamento: Centro
   - Margem superior: 8px

7. Botão Principal:
   - Margem superior: 32px da nota
   - Largura: 100%
   - Altura: 48px
   - Background: #6750A4
   - Texto: "VOLTAR AO LOGIN"
   - Cor texto: Branco
   - Font: Inter SemiBold, 14px
   - Border radius: 12px

8. Botão Secundário:
   - Margem superior: 12px do botão principal
   - Largura: 100%
   - Altura: 48px
   - Background: Transparente
   - Border: 1px #6750A4
   - Texto: "ENVIAR NOVAMENTE"
   - Cor texto: #6750A4
   - Font: Inter SemiBold, 14px
   - Border radius: 12px

ESTILO:
- Estado de sucesso claro e positivo
- Ícone verde transmite conclusão
- Hierarquia visual guia para próxima ação
- Opção secundária para reenvio

Parte 6: Assets e Recursos

6.1. Ícones Necessários

Ícones integrados do Flutter (Material Icons):

- Email: `Icons.email_outlined`
- Senha: `Icons.lock_outline`
- Visibilidade: `Icons.visibility` / `Icons.visibility_off`
- Usuário: `Icons.person_outline`
- Check: `Icons.check_circle`
- Erro: `Icons.error_outline`
- Info: `Icons.info_outline`
- Voltar: `Icons.arrow_back`
- Fechar: `Icons.close`

Logos externos (adicionar em `assets/icons/`):

- Google logo: `google_logo.png` (24x24px)
- GitHub logo: `github_logo.png` (24x24px)
- Facebook logo: `facebook_logo.png` (24x24px)

Você pode baixar logos oficiais em:

- Google: Google Brand Guidelines
- GitHub: GitHub Logos
- Facebook: Meta Brand Resources

6.2. Imagens e Ilustrações

Para telas de estado vazio ou onboarding, considere usar:

- Undraw - Ilustrações SVG gratuitas e customizáveis
- DrawKit - Ilustrações vetoriais gratuitas
- Storyset - Ilustrações animadas

6.3. Fontes

Google Fonts recomendadas:

```yaml
```

```
# pubspec.yaml
dependencies:
  google_fonts: ^6.0.0
```

Opções:

1. **Inter** - Moderna, clean, excelente legibilidade
2. **Roboto** - Padrão Material, segura
3. **Poppins** - Arredondada, amigável
4. **Montserrat** - Geométrica, profissional

```dart
// Exemplo de uso
import 'package:google_fonts/google_fonts.dart';

TextStyle headlineStyle = GoogleFonts.inter(
  fontSize: 24,
  fontWeight: FontWeight.w600,
);
```

Parte 7: Animações e Microinterações

7.1. Transições entre Telas

```dart
```

```dart
// Transição suave entre login e cadastro
Navigator.of(context).push(
  PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => SignUpScreen(),
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      const begin = Offset(1.0, 0.0);
      const end = Offset.zero;
      const curve = Curves.easeInOut;

      var tween = Tween(begin: begin, end: end).chain(
        CurveTween(curve: curve),
      );

      return SlideTransition(
        position: animation.drive(tween),
        child: child,
      );
    },
    transitionDuration: const Duration(milliseconds: 300),
  ),
);
```

7.2. Animação de Loading no Botão

```dart
// Já implementado no AuthButton
// Quando isLoading = true, mostra CircularProgressIndicator
```

7.3. Shake Animation para Erro

```dart
```

```dart
// lib/widgets/shake_widget.dart

import 'package:flutter/material.dart';

class ShakeWidget extends StatefulWidget {
  final Widget child;
  final bool shake;

  const ShakeWidget({
    super.key,
    required this.child,
    required this.shake,
  });

  @override
  State<ShakeWidget> createState() => _ShakeWidgetState();
}

class _ShakeWidgetState extends State<ShakeWidget>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;
  late Animation<double> _animation;

  @override
  void initState() {
    super.initState();
    _controller = AnimationController(
      duration: const Duration(milliseconds: 500),
      vsync: this,
    );

    _animation = Tween<double>(begin: 0, end: 10).animate(
      CurvedAnimation(
        parent: _controller,
        curve: Curves.elasticIn,
      ),
    );
  }

  @override
  void didUpdateWidget(ShakeWidget oldWidget) {
    super.didUpdateWidget(oldWidget);
    if (widget.shake && !oldWidget.shake) {
      _controller.forward(from: 0);
    }
  }
```

```dart
  @override
  void dispose() {
    _controller.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return AnimatedBuilder(
      animation: _animation,
      builder: (context, child) {
        return Transform.translate(
          offset: Offset(_animation.value, 0),
          child: child,
        );
      },
      child: widget.child,
    );
  }
}

// Uso:
ShakeWidget(
  shake: hasError,
  child: CustomTextField(...),
)
```

7.4. Fade In para Mensagens de Sucesso

dart

```dart
AnimatedOpacity(
  opacity: isVisible ? 1.0 : 0.0,
  duration: const Duration(milliseconds: 300),
  child: Text('Login realizado com sucesso!'),
)
```

Conclusão da Seção

Nesta seção, você aprendeu:

✅ **Princípios de design** específicos para autenticação
✅ **Material Design 3** aplicado completamente (cores, tipografia, componentes)

✅ **ThemeData completo** para light e dark themes
✅ **Componentes reutilizáveis** prontos para implementação
✅ **Especificações detalhadas** de cada tela com medidas exatas
✅ **Prompts para IA** que você pode usar para gerar protótipos
✅ **Assets e recursos** necessários
✅ **Animações** para melhorar a experiência

Como usar este conhecimento:

1. **Codificando do zero**: Use os componentes prontos e o ThemeData para criar suas telas
2. **Gerando com IA**: Use os prompts fornecidos em ferramentas como Midjourney, Figma AI, v0.dev
3. **Protótipos rápidos**: Use as especificações para criar mockups no Figma
4. **Referência**: Consulte as medidas e cores sempre que precisar

Próximos passos:

Na próxima seção, você terá **desafios práticos** para aplicar tudo que aprendeu, com critérios claros de avaliação!

Fim da Seção 12: Protótipos Visuais e Material Design 3