**CURRENCY CONVERTOR**

**PROJECT GROUP MEMBERS**

|  |  |
| --- | --- |
| **Students Id** | **Students Name** |
| ***Student1426616*** | ***Syed Muhammad Zain Sarfraz*** |
| ***Student1450998*** | ***Huda Shahid*** |
| ***Student1424649*** | ***Tayyaba Muslim*** |
| ***Student1452427*** | ***Jaisha Jaseem*** |
| ***Student1421636*** | ***Muhammad Anas*** |
| ***Student1407243*** | ***Muhammad Yasir*** |

**FACULTY**

**SIR HARIS NASEER**

**CO-ORDINATOR**

**MISS MAHA HAIDER**

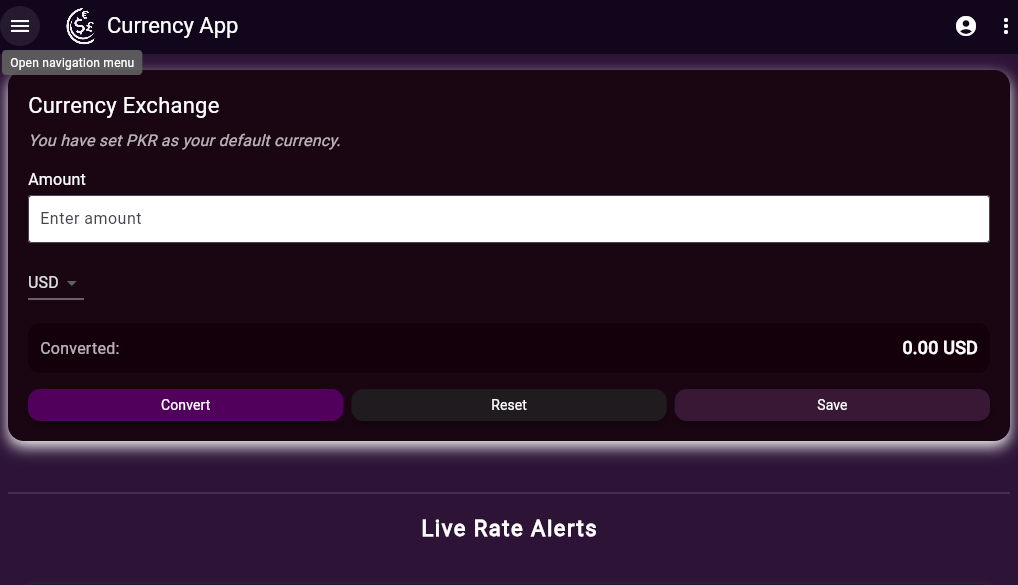
**CURRICULUM**

**7062**

**BATCH**

**2302C2**

**HOME PAGE**



class HomeScreen extends StatefulWidget {

  static String routeName = '/home';

  const HomeScreen({Key? key}) : super(key: key);

  @override

  State<HomeScreen> createState() => \_HomeScreenState();

}

class \_HomeScreenState extends State<HomeScreen> {

  final TextEditingController \_sendController = TextEditingController();

  double \_convertedAmount = 0.0;

  double \_exchangeRate = 0.0036;

  String \_defaultCurrency = 'PKR';

  String \_selectedCurrency = 'USD';

  String? \_userId;

  bool \_isLoading = true;

  final int \_itemsPerPage = 3; // Adjust based on your layout

  int \_currentPage = 0;

  @override

  void initState() {

    super.initState();

    \_userId = FirebaseAuth.instance.currentUser?.uid;

    \_loadUserSettings();

    \_listenToCurrencyChanges();

    Future.delayed(Duration(seconds: 2), () {

      if (mounted) {

        setState(() => \_isLoading = false); // ✅ Safe now

      }

    });

  }

  void \_listenToCurrencyChanges() {

    if (\_userId != null) {

      FirebaseFirestore.instance

          .collection('app\_settings')

          .doc(\_userId)

          .snapshots()

          .listen((snapshot) {

        if (snapshot.exists) {

          final data = snapshot.data() as Map<String, dynamic>?;

          if (data != null) {

            final newCurrency = data['defaultCurrency'] ?? 'USD';

            if (newCurrency != \_defaultCurrency) {

              setState(() {

                \_defaultCurrency = newCurrency;

                \_selectedCurrency = newCurrency;

                \_exchangeRate =

                    \_getExchangeRate(\_defaultCurrency, \_selectedCurrency);

              });

            }

          }

        }

      });

    }

  }

  void \_loadUserSettings() async {

    if (\_userId != null) {

      final doc = await FirebaseFirestore.instance

          .collection('app\_settings')

          .doc(\_userId)

          .get();

      if (doc.exists) {

        final settings = doc.data()?['settings'] ?? {};

        setState(() {

          \_defaultCurrency = settings['defaultCurrency'] ?? 'PKR';

          \_selectedCurrency = \_defaultCurrency;

          \_exchangeRate = \_getExchangeRate(\_defaultCurrency, \_selectedCurrency);

        });

      }

    }

  }

  double \_getExchangeRate(String from, String to) {

    const rates = {

      'PKR->USD': 0.0036,

      'PKR->EUR': 0.0033,

      'USD->PKR': 277.0,

      'EUR->PKR': 303.0,

      'USD->EUR': 0.92,

      'EUR->USD': 1.09,

    };

    return rates['$from->$to'] ?? 1.0;

  }

  void \_convertCurrency() {

    double sendAmount = double.tryParse(\_sendController.text) ?? 0.0;

    double rate = \_getExchangeRate(\_defaultCurrency, \_selectedCurrency);

    setState(() {

      \_exchangeRate = rate;

      \_convertedAmount = sendAmount \* rate;

    });

  }

  void \_resetConversion() {

    setState(() {

      \_sendController.clear();

      \_convertedAmount = 0.0;

      \_selectedCurrency = \_defaultCurrency;

    });

  }

  void \_saveConversion() async {

    // Check if the user is logged in (userId is not null)

    if (\_userId == null) {

      // If userId is null, navigate to the login screen

      Navigator.pushNamed(context, EmailPasswordLogin.routeName);

      return; // Return early to stop further execution

    }

    // Proceed with saving the conversion

    final double sendAmount = double.tryParse(\_sendController.text) ?? 0.0;

    if (sendAmount == 0.0) {

      \_showErrorAlert('Please enter an amount greater than 0.');

      return;

    }

    final double receiveAmount = sendAmount \* \_exchangeRate;

    final existingConversion = await FirebaseFirestore.instance

        .collection('saved\_conversions')

        .where('userId', isEqualTo: \_userId)

        .where('defaultCurrency', isEqualTo: \_defaultCurrency)

        .where('convertedCurrency', isEqualTo: \_selectedCurrency)

        .where('convertedAmount', isEqualTo: receiveAmount)

        .get();

    if (existingConversion.docs.isNotEmpty) {

      \_showErrorAlert('This conversion has already been saved!');

      return;

    }

    final savedConversion = SavedConversion(

      id: DateTime.now().toIso8601String(),

      userId: \_userId!,

      defaultCurrency: \_defaultCurrency,

      convertedCurrency: \_selectedCurrency,

      convertedAmount: receiveAmount,

      originalAmount: sendAmount,

    );

    await FirebaseFirestore.instance.collection('saved\_conversions').add({

      ...savedConversion.toMap(),

      'createdAt': Timestamp.now(),

      'userId': \_userId,

    });

    \_showSuccessAlert('Conversion saved successfully!');

  }

  void \_showErrorAlert(String message) {

    ScaffoldMessenger.of(context).showSnackBar(

      SnackBar(content: Text(message), backgroundColor: Colors.red),

    );

  }

  void \_showSuccessAlert(String message) {

    ScaffoldMessenger.of(context).showSnackBar(

      SnackBar(content: Text(message), backgroundColor: Colors.green),

    );

  }

  Widget \_buildConversionBox() {

    return Container(

      margin: const EdgeInsets.only(bottom: 20),

      padding: const EdgeInsets.all(20),

      decoration: BoxDecoration(

        color: const Color.fromARGB(255, 26, 5, 19),

        borderRadius: BorderRadius.circular(16),

        boxShadow: const [

          BoxShadow(

            color: Color.fromARGB(255, 250, 250, 250),

            blurRadius: 12,

            offset: Offset(0, 6),

          )

        ],

      ),

      child: Column(

        crossAxisAlignment: CrossAxisAlignment.start,

        children: [

          const Text('Currency Exchange',

              style: TextStyle(fontSize: 22, color: Colors.white)),

          const SizedBox(height: 8),

          Text('You have set $\_defaultCurrency as your default currency.',

              style: const TextStyle(

                  fontSize: 16,

                  color: Colors.white70,

                  fontStyle: FontStyle.italic)),

          const SizedBox(height: 16),

          const Text('Amount',

              style: TextStyle(color: Colors.white, fontSize: 16)),

          const SizedBox(height: 4),

          TextFormField(

            controller: \_sendController,

            keyboardType: TextInputType.number,

            decoration: const InputDecoration(

              filled: true,

              fillColor: Colors.white,

              hintText: 'Enter amount',

              border: OutlineInputBorder(),

            ),

          ),

          const SizedBox(height: 16),

          DropdownButton<String>(

            dropdownColor: const Color.fromARGB(255, 45, 20, 55),

            value: \_selectedCurrency,

            onChanged: (value) {

              if (value != null) {

                setState(() {

                  \_selectedCurrency = value;

                  \_exchangeRate =

                      \_getExchangeRate(\_defaultCurrency, \_selectedCurrency);

                });

              }

            },

            items: [

              'USD',

              'PKR',

              'EUR',

              'GBP',

              'INR',

              'CAD',

              'AUD',

              'CNY',

              'JPY'

            ].map((currency) {

              return DropdownMenuItem(

                value: currency,

                child: Text(currency,

                    style: TextStyle(

                      color: currency == \_defaultCurrency

                          ? Colors.white

                          : Colors.grey[300],

                    )),

              );

            }).toList(),

          ),

          const SizedBox(height: 16),

          Container(

            padding: const EdgeInsets.all(12),

            decoration: BoxDecoration(

              color: const Color.fromARGB(255, 20, 0, 11),

              borderRadius: BorderRadius.circular(12),

            ),

            child: Row(

              mainAxisAlignment: MainAxisAlignment.spaceBetween,

              children: [

                const Text('Converted:',

                    style: TextStyle(color: Colors.white70, fontSize: 16)),

                Text(

                  '${\_convertedAmount.toStringAsFixed(2)} $\_selectedCurrency',

                  style: const TextStyle(

                      fontWeight: FontWeight.bold,

                      color: Colors.white,

                      fontSize: 18),

                ),

              ],

            ),

          ),

          const SizedBox(height: 16),

          Row(

            children: [

              \_buildActionButton('Convert', \_convertCurrency,

                  const Color.fromARGB(255, 81, 0, 92)),

              const SizedBox(width: 8),

              \_buildActionButton('Reset', \_resetConversion,

                  const Color.fromARGB(255, 31, 27, 30)),

              const SizedBox(width: 8),

              \_buildActionButton('Save', \_saveConversion,

                  const Color.fromARGB(255, 56, 24, 53)),

            ],

          )

        ],

      ),

    );

  }

  Widget \_buildActionButton(String text, VoidCallback onTap, Color color) {

    return Expanded(

      child: ElevatedButton(

        onPressed: onTap,

        style: ElevatedButton.styleFrom(

          backgroundColor: color,

          foregroundColor: Colors.white,

          elevation: 6,

          shadowColor: Colors.black,

          shape:

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

        ),

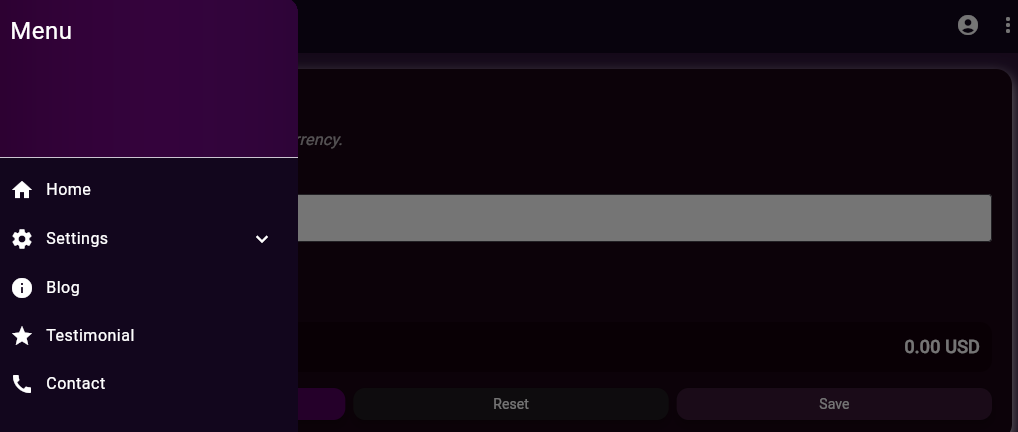
        child: Text(text),

      ),

    );

  }

**Navbar**

****

 @override

  Widget build(BuildContext context) {

    final user = FirebaseAuth.instance.currentUser;

    return PageWithLoader(

      isLoading: \_isLoading,

      child: Scaffold(

        appBar: AppBar(

          backgroundColor: const Color.fromARGB(255, 18, 6, 29),

          iconTheme: const IconThemeData(color: Colors.white),

          title: Row(

            children: [

              Image.asset(

                'assets/logo.png',

                height: 40,

              ),

              const SizedBox(width: 10),

              const Text(

                'Currency App',

                style: TextStyle(color: Colors.white),

              ),

            ],

          ),

          actions: [

            IconButton(

              icon: const Icon(Icons.account\_circle, color: Colors.white),

              onPressed: () {

                final user = FirebaseAuth.instance.currentUser;

                if (user != null) {

                  \_showProfileDialog(

                      context, user); // Show profile if logged in

                } else {

                  Navigator.pushNamed(

                      context,

                      EmailPasswordLogin

                          .routeName); // Show login screen if not logged in

                }

              },

            ),

            PopupMenuButton<String>(

              icon: const Icon(Icons.more\_vert, color: Colors.white),

              onSelected: (value) async {

                if (value == 'signout') {

                  await context.read<FirebaseAuthMethods>().signOut(context);

                  Navigator.pushAndRemoveUntil(

                    context,

                    MaterialPageRoute(

                        builder: (\_) => const EmailPasswordLogin()),

                    (route) => false,

                  );

                } else if (value == 'delete') {

                  await context

                      .read<FirebaseAuthMethods>()

                      .deleteAccount(context);

                } else if (value == 'signup') {

                  Navigator.pushNamed(

                      context,

                      EmailPasswordSignup

                          .routeName); // Navigate to signup screen

                } else if (value == 'login') {

                  Navigator.pushNamed(context,

                      EmailPasswordLogin.routeName); // Navigate to login screen

                }

              },

              itemBuilder: (context) {

                final user = FirebaseAuth.instance.currentUser;

                // If the user is not logged in, show login/signup options

                if (user == null) {

                  return [

                    const PopupMenuItem(

                        value: 'signup', child: Text('Sign Up')),

                    const PopupMenuItem(value: 'login', child: Text('Login')),

                  ];

                } else {

                  // If the user is logged in, show sign out/delete options

                  return const [

                    PopupMenuItem(value: 'signout', child: Text('Sign Out')),

                    PopupMenuItem(

                        value: 'delete', child: Text('Delete Account')),

                  ];

                }

              },

            ),

Widget \_buildDrawer() {

    return Drawer(

      backgroundColor: const Color.fromARGB(255, 18, 6, 29),

      child: ListView(

        padding: EdgeInsets.zero,

        children: [

          const DrawerHeader(

            decoration: BoxDecoration(

              gradient: LinearGradient(

                colors: [

                  Color.fromARGB(255, 44, 1, 49),

                  Color.fromARGB(99, 89, 2, 97)

                ],

              ),

            ),

            child: Text('Menu',

                style: TextStyle(color: Colors.white, fontSize: 24)),

          ),

          ListTile(

            leading: const Icon(Icons.home, color: Colors.white),

            title: const Text('Home', style: TextStyle(color: Colors.white)),

            onTap: () {

              Navigator.pop(context);

              Navigator.pushReplacement(

                context,

                MaterialPageRoute(builder: (\_) => const HomeScreen()),

              );

            },

          ),

          ExpansionTile(

            leading: const Icon(Icons.settings, color: Colors.white),

            title:

                const Text('Settings', style: TextStyle(color: Colors.white)),

            iconColor: Colors.white,

            collapsedIconColor: Colors.white,

            children: [

              // Show "Rate Alerts," "Currency News," and "Help Center" if the user is logged out (user == null)

              if (FirebaseAuth.instance.currentUser == null ||

                  FirebaseAuth.instance.currentUser != null)

                \_buildDrawerItem(

                  icon: Icons.notifications\_active,

                  text: 'Rate Alerts',

                  screen: const RateAlertsScreen(),

                ),

              if (FirebaseAuth.instance.currentUser == null ||

                  FirebaseAuth.instance.currentUser != null)

                \_buildDrawerItem(

                  icon: Icons.trending\_up,

                  text: 'Currency News',

                  screen: const CurrencyNewsScreen(),

                ),

              if (FirebaseAuth.instance.currentUser == null ||

                  FirebaseAuth.instance.currentUser != null)

                \_buildDrawerItem(

                  icon: Icons.support\_agent,

                  text: 'Help Center',

                  screen: const HelpCenterScreen(),

                ),

              // Show "Conversion History" and "Default Currency" if the user is logged in (user != null)

              if (FirebaseAuth.instance.currentUser != null)

                \_buildDrawerItem(

                  icon: Icons.history,

                  text: 'Conversion History',

                  screen: const ConversionHistoryScreen(),

                ),

              if (FirebaseAuth.instance.currentUser != null)

                \_buildDrawerItem(

                  icon: Icons.currency\_exchange,

                  text: 'Default Currency',

                  screen: const DefaultCurrencyScreen(),

                ),

            ],

          ),

          ListTile(

              leading: const Icon(Icons.info, color: Colors.white),

              title: const Text('Blog', style: TextStyle(color: Colors.white)),

              onTap: () {

               Navigator.pop(context);

          Navigator.pushReplacement(

      context,

  MaterialPageRoute(

    builder: (\_) => BlogScreen(post: BlogScreen.blogPosts[0]),

  ),

  );

              }),

          ListTile(

            leading: const Icon(Icons.star, color: Colors.white),

            title: const Text('Testimonial',

                style: TextStyle(color: Colors.white)),

            onTap: () {

              Navigator.pop(context);

              Navigator.pushReplacement(

                context,

                MaterialPageRoute(builder: (\_) => const TestimonialPage()),

              );

            },

          ),

          ListTile(

              leading: const Icon(Icons.call, color: Colors.white),

              title:

                  const Text('Contact', style: TextStyle(color: Colors.white)),

              onTap: () {

                Navigator.pop(context);

                Navigator.pushReplacement(

                  context,

                  MaterialPageRoute(builder: (\_) => const ContactPage()),

                );

              }),

        ],

      ),

    );

  }

  Widget \_buildDrawerItem(

      {required IconData icon, required String text, required Widget screen}) {

    return ListTile(

      leading: Icon(icon, color: Colors.white70),

      title: Text(text, style: const TextStyle(color: Colors.white70)),

      onTap: () {

        Navigator.pop(context);

        Navigator.push(context, MaterialPageRoute(builder: (\_) => screen));

      },

    );

  }

  void \_showProfileDialog(BuildContext context, User user) {

    showDialog(

      context: context,

      builder: (\_) {

        return AlertDialog(

          title: const Text('Profile Details'),

          content: Column(

            mainAxisSize: MainAxisSize.min,

            children: [

              if (!user.isAnonymous && user.phoneNumber == null)

                Text('Email: ${user.email}'),

              if (user.phoneNumber != null) Text('Phone: ${user.phoneNumber}'),

              // Text('UID: ${user.uid}'),

              if (!user.emailVerified && !user.isAnonymous)

                CustomButton(

                  label: 'Verify Email',

                  onPressed: () {

                    context

                        .read<FirebaseAuthMethods>()

                        .sendEmailVerification(context);

                  },

                ),

            ],

          ),

          actions: [

            TextButton(

              onPressed: () => Navigator.pop(context),

              child: const Text('Close'),

            ),

          ],

        );

      },

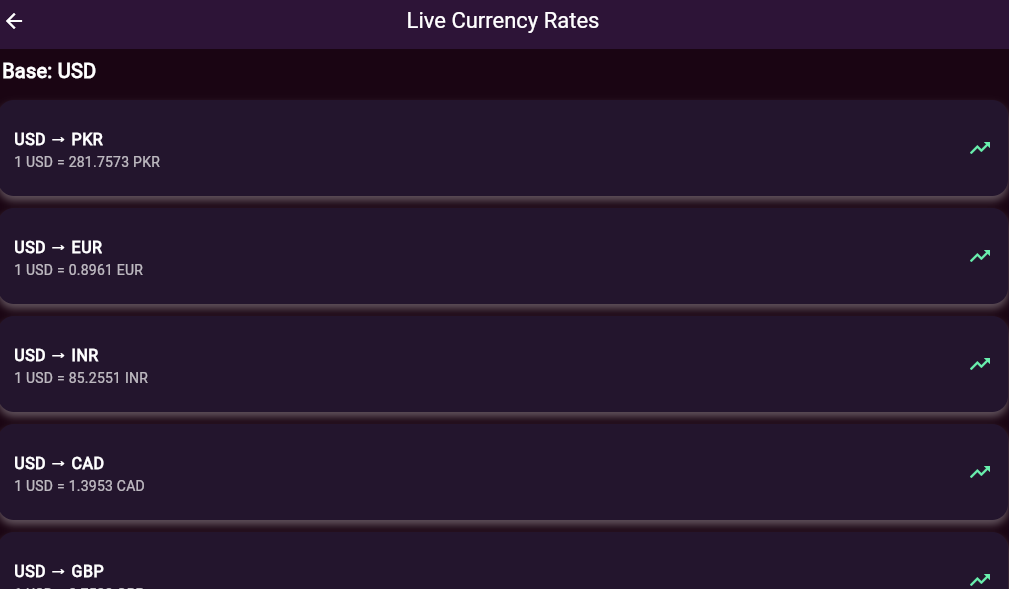
    );

  }

}

**Settings**

**Rate Alerts**

****

class RateAlertsScreen extends StatefulWidget {

  static const String routeName = '/rate-alerts';

  const RateAlertsScreen({Key? key}) : super(key: key);

  @override

  State<RateAlertsScreen> createState() => \_RateAlertsScreenState();

}

class \_RateAlertsScreenState extends State<RateAlertsScreen> {

  bool \_isLoading = true;

  bool \_disposed = false;

  final Color bgColor = const Color.fromARGB(255, 26, 5, 19);

  final Color cardColor = const Color.fromARGB(255, 35, 21, 45);

  final List<String> currencies = [

    'USD',

    'PKR',

    'EUR',

    'INR',

    'CAD',

    'GBP',

    'AUD',

    'CNY',

    'JPY'

  ];

  final Map<String, Map<String, double>> \_allRates = {};

  @override

  void initState() {

    super.initState();

    \_fetchAllRates();

  }

  void dispose() {

    \_disposed = true;

    super.dispose();

  }

  Future<void> \_fetchAllRates() async {

    const String apiKey =

        '03a85ceec4f1b8e243e2c87e'; // no leading/trailing spaces

    try {

      for (String base in currencies) {

        final url = Uri.parse(

            'https://v6.exchangerate-api.com/v6/03a85ceec4f1b8e243e2c87e/latest/USD');

        final response = await http.get(url);

        if (response.statusCode == 200) {

          final data = jsonDecode(response.body);

          if (data['result'] == 'success' && data['conversion\_rates'] != null) {

            final rawRates =

                Map<String, dynamic>.from(data['conversion\_rates']);

            final Map<String, double> filteredRates = {};

            for (String target in currencies) {

              if (target != base && rawRates.containsKey(target)) {

                filteredRates[target] =

                    double.tryParse(rawRates[target].toString()) ?? 0.0;

              }

            }

            if (mounted && !\_disposed) {

              setState(() {

                \_allRates[base] = filteredRates;

              });

            }

          } else {

            print(

                'API error for $base: ${data['error-type'] ?? 'Unexpected response'}');

          }

        } else {

          print('HTTP error for $base: ${response.statusCode}');

        }

      }

    } catch (e) {

      print('Error fetching rates: $e');

    }

    if (mounted && !\_disposed) {

      setState(() => \_isLoading = false);

    }

  }

  Widget \_buildRateSection(String base, Map<String, double> rates) {

    return Column(

      crossAxisAlignment: CrossAxisAlignment.start,

      children: [

        Padding(

          padding: const EdgeInsets.symmetric(horizontal: 16, vertical: 8),

          child: Text(

            'Base: $base',

            style: const TextStyle(

              color: Colors.white,

              fontSize: 20,

              fontWeight: FontWeight.bold,

            ),

          ),

        ),

        ...rates.entries

            .map((entry) => \_buildRateCard(base, entry.key, entry.value))

            .toList(),

        const Divider(color: Colors.white24, thickness: 1, height: 32),

      ],

    );

  }

  Widget \_buildRateCard(String from, String to, double rate) {

    return Container(

      margin: const EdgeInsets.symmetric(vertical: 6, horizontal: 12),

      decoration: BoxDecoration(

        color: cardColor,

        borderRadius: BorderRadius.circular(16),

        boxShadow: [

          BoxShadow(

            color: const Color.fromARGB(255, 255, 255, 255).withOpacity(0.3),

            offset: const Offset(0, 5),

            blurRadius: 6,

          ),

        ],

      ),

      child: ListTile(

        contentPadding: const EdgeInsets.all(16),

        title: Text(

          '$from → $to',

          style: const TextStyle(

            fontSize: 16,

            fontWeight: FontWeight.bold,

            color: Colors.white,

          ),

        ),

        subtitle: Text(

          '1 $from = ${rate.toStringAsFixed(4)} $to',

          style: const TextStyle(color: Colors.white70),

        ),

        trailing: const Icon(Icons.trending\_up, color: Colors.greenAccent),

      ),

    );

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      backgroundColor: bgColor,

      appBar: AppBar(

        title: const Text('Live Currency Rates'),

        centerTitle: true,

        backgroundColor: const Color.fromARGB(255, 45, 20, 55),

        foregroundColor: Colors.white,

      ),

      body: PageWithLoader(

        key: const ValueKey('rate\_alerts\_loader'),

        isLoading: \_isLoading,

        child: \_allRates.isEmpty

            ? const Center(

                child: Text(

                  'No rates available.',

                  style: TextStyle(color: Colors.white70),

                ),

              )

            : ListView(

                children: \_allRates.entries

                    .map((entry) => \_buildRateSection(entry.key, entry.value))

                    .toList(),

              ),

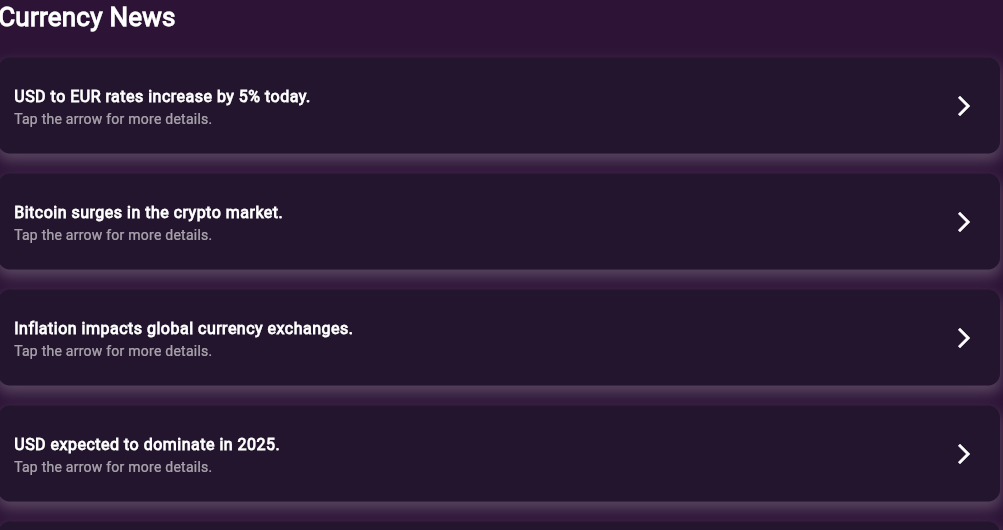
      ),

    );

  }

}

**Currency News**

****

class CurrencyNewsScreen extends StatelessWidget {

  static const String routeName = '/currency-news';

  const CurrencyNewsScreen({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    final List<Map<String, String>> newsArticles = [

      {

        'title': 'USD to EUR rates increase by 5% today.',

        'details':

            'USD strengthened significantly against the Euro, reaching a 5% increase. Current rate: 1 USD = 0.94 EUR.',

      },

      {

        'title': 'Bitcoin surges in the crypto market.',

        'details':

            'Bitcoin sees a 7% surge in 24 hours. Current value: \$41,200. Analysts cite ETF approval as the cause.',

      },

      {

        'title': 'Inflation impacts global currency exchanges.',

        'details':

            'Rising inflation rates have caused fluctuations in emerging market currencies, especially in Asia.',

      },

      {

        'title': 'USD expected to dominate in 2025.',

        'details':

            'Financial experts suggest that a strong US economy will boost the dollar\'s dominance worldwide.',

      },

      {

        'title': 'Euro hits a 6-month high.',

        'details':

            'The Euro strengthens due to positive EU economic forecasts. Current rate: 1 EUR = 1.10 USD.',

      },

    ];

    return Column(

      crossAxisAlignment: CrossAxisAlignment.start,

      children: [

        const Text(

          'Currency News',

          style: TextStyle(

            color: Colors.white,

            fontSize: 26,

            fontWeight: FontWeight.bold,

          ),

        ),

        const SizedBox(height: 12),

        ListView.builder(

          shrinkWrap: true,

          physics: const NeverScrollableScrollPhysics(),

          itemCount: newsArticles.length,

          itemBuilder: (context, index) {

            final article = newsArticles[index];

            return Container(

              margin: const EdgeInsets.symmetric(vertical: 10),

              decoration: BoxDecoration(

                color: const Color.fromARGB(255, 35, 21, 45),

                borderRadius: BorderRadius.circular(12),

                boxShadow: [

                  BoxShadow(

                    color: Colors.white.withOpacity(0.2),

                    offset: const Offset(0, 6),

                    blurRadius: 10,

                    spreadRadius: 1,

                  ),

                ],

              ),

              child: ListTile(

                contentPadding: const EdgeInsets.all(16),

                title: Text(

                  article['title']!,

                  style: const TextStyle(

                    color: Colors.white,

                    fontWeight: FontWeight.w600,

                    fontSize: 16,

                  ),

                ),

                subtitle: const Text(

                  'Tap the arrow for more details.',

                  style: TextStyle(color: Colors.white60),

                ),

                trailing: IconButton(

                  icon:

                      const Icon(Icons.arrow\_forward\_ios, color: Colors.white),

                  onPressed: () {

                    showDialog(

                      context: context,

                      builder: (context) => AlertDialog(

                        backgroundColor: const Color(0xFF1E1B38),

                        shape: RoundedRectangleBorder(

                            borderRadius: BorderRadius.circular(12)),

                        title: Text(

                          'Details',

                          style: TextStyle(

                              color: Colors.tealAccent[100],

                              fontWeight: FontWeight.bold),

                        ),

                        content: Text(

                          article['details']!,

                          style: const TextStyle(color: Colors.white),

                        ),

                        actions: [

                          TextButton(

                            onPressed: () => Navigator.pop(context),

                            child: const Text('Close',

                                style: TextStyle(color: Colors.tealAccent)),

                          ),

                        ],

                      ),

                    );

                  },

                ),

              ),

            );

          },

        ),

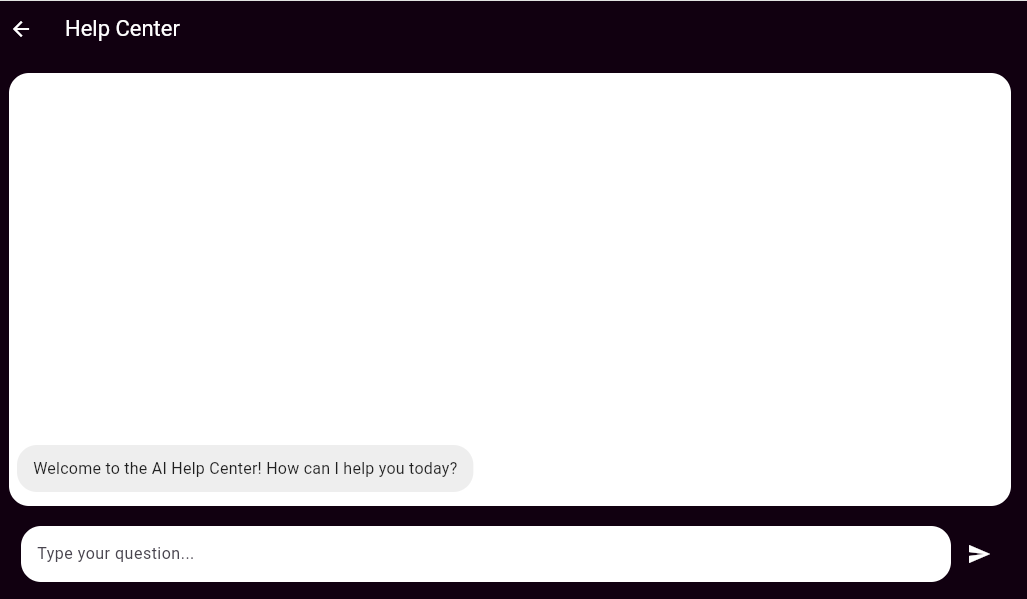
      ],

    );

  }

}

**Help Center**

****

class HelpCenterScreen extends StatefulWidget {

  static const String routeName = '/help-center';

  const HelpCenterScreen({Key? key}) : super(key: key);

  @override

  State<HelpCenterScreen> createState() => \_HelpCenterScreenState();

}

class \_HelpCenterScreenState extends State<HelpCenterScreen> {

  final TextEditingController \_controller = TextEditingController();

  final List<Map<String, String>> \_messages = [

    {

      'sender': 'bot',

      'text': 'Welcome to the AI Help Center! How can I help you today?'

    }

  ];

  final Color primaryColor = const Color.fromARGB(255, 17, 0, 16);

  bool \_isTyping = false;

  // Function to send the message and get the AI response

  Future<void> \_sendMessage(String text) async {

    if (text.trim().isEmpty) return;

    // Add user's message

    setState(() {

      \_messages.insert(0, {'sender': 'user', 'text': text.trim()});

      \_controller.clear();

      \_isTyping = true;

    });

    try {

      final botReply = await \_getBotResponse(text.trim());

      setState(() {

        \_messages.insert(0, {'sender': 'bot', 'text': botReply});

      });

    } catch (e) {

      setState(() {

        \_messages.insert(0,

            {'sender': 'bot', 'text': '❗ Error occurred. Please try again.'});

      });

    } finally {

      setState(() {

        \_isTyping = false;

      });

    }

  }

  // Function to get the AI response from OpenAI

  Future<String> \_getBotResponse(String query) async {

    const apiKey = 'AIzaSyB3NM76djkuDnJFCdNHWG2xOxN3r-sbRIo';

    final apiUrl =

        'https://generativelanguage.googleapis.com/v1beta/models/gemini-2.0-flash:generateContent?key=$apiKey';

    final response = await http.post(

      Uri.parse(apiUrl),

      headers: {

        'Content-Type': 'application/json',

      },

      body: jsonEncode({

        "contents": [

          {

            "parts": [

              {"text": query}

            ],

            "role": "user"

          }

        ]

      }),

    );

    if (response.statusCode == 200) {

      final data = jsonDecode(response.body);

      final botMessage = data['candidates'][0]['content']['parts'][0]['text'];

      return botMessage.trim();

    } else {

      print("Error: ${response.body}");

      throw Exception('Failed to fetch AI reply');

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        backgroundColor: primaryColor,

        title: const Text('Help Center'),

        foregroundColor: Colors.white,

      ),

      body: Container(

        color: primaryColor,

        padding: const EdgeInsets.all(16),

        child: Column(

          children: [

            Expanded(

              child: Container(

                decoration: BoxDecoration(

                  color: Colors.white,

                  borderRadius: BorderRadius.circular(20),

                ),

                child: ListView.builder(

                  reverse: true,

                  padding: const EdgeInsets.all(8),

                  itemCount: \_messages.length + (\_isTyping ? 1 : 0),

                  itemBuilder: (context, index) {

                    if (\_isTyping && index == 0) {

                      return \_buildTypingIndicator();

                    }

                    final message = \_messages[\_isTyping ? index - 1 : index];

                    final isBot = message['sender'] == 'bot';

                    return Align(

                      alignment:

                          isBot ? Alignment.centerLeft : Alignment.centerRight,

                      child: Container(

                        margin: const EdgeInsets.symmetric(vertical: 6),

                        padding: const EdgeInsets.symmetric(

                            horizontal: 16, vertical: 12),

                        decoration: BoxDecoration(

                          color: isBot

                              ? Colors.grey[200]

                              : primaryColor.withOpacity(0.9),

                          borderRadius: BorderRadius.circular(20),

                        ),

                        child: Text(

                          message['text']!,

                          style: TextStyle(

                            color: isBot ? Colors.black87 : Colors.white,

                            fontSize: 16,

                          ),

                        ),

                      ),

                    );

                  },

                ),

              ),

            ),

            const SizedBox(height: 12),

            Container(

              padding: const EdgeInsets.symmetric(horizontal: 12, vertical: 8),

              decoration: BoxDecoration(

                color: primaryColor,

                borderRadius: BorderRadius.circular(20),

              ),

              child: Row(

                children: [

                  Expanded(

                    child: Container(

                      padding: const EdgeInsets.symmetric(

                          horizontal: 16, vertical: 4),

                      decoration: BoxDecoration(

                        color: Colors.white,

                        borderRadius: BorderRadius.circular(20),

                      ),

                      child: TextField(

                        controller: \_controller,

                        decoration: const InputDecoration(

                          hintText: 'Type your question...',

                          border: InputBorder.none,

                        ),

                        onSubmitted: (value) => \_sendMessage(value),

                      ),

                    ),

                  ),

                  const SizedBox(width: 8),

                  CircleAvatar(

                    backgroundColor: primaryColor,

                    child: IconButton(

                      icon: const Icon(Icons.send, color: Colors.white),

                      onPressed: () => \_sendMessage(\_controller.text),

                    ),

                  ),

                ],

              ),

            ),

          ],

        ),

      ),

    );

  }

  // Typing indicator widget (Spinner)

  Widget \_buildTypingIndicator() {

    return Align(

      alignment: Alignment.centerLeft,

      child: Container(

        margin: const EdgeInsets.symmetric(vertical: 6),

        padding: const EdgeInsets.symmetric(horizontal: 16, vertical: 12),

        decoration: BoxDecoration(

          color: Colors.grey[300],

          borderRadius: BorderRadius.circular(20),

        ),

        child: const SpinKitThreeBounce(

          color: Colors.black87,

          size: 20.0,

        ),

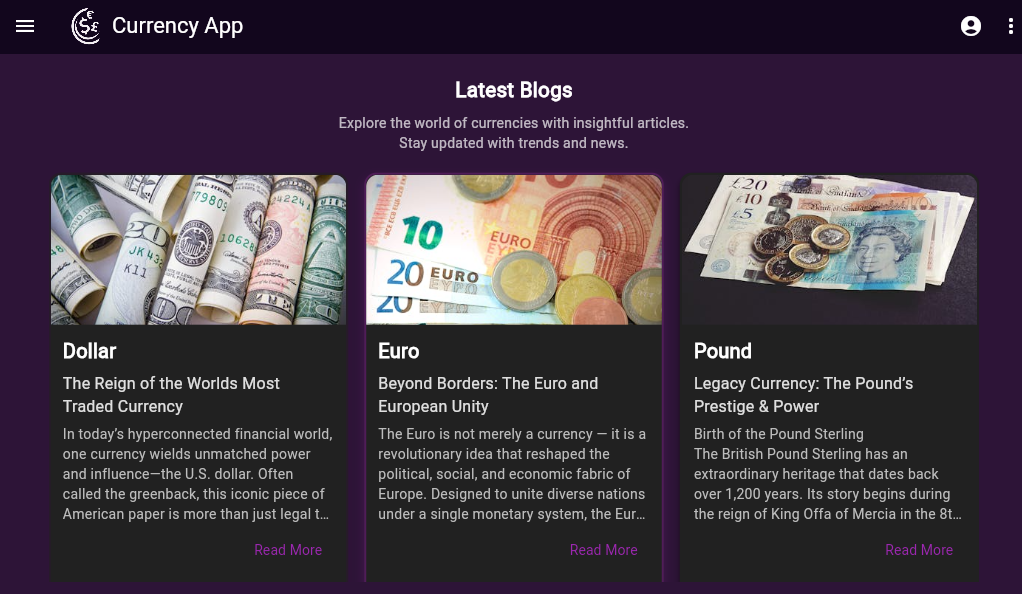
      ),

    );

  }

}

**Blog**

****

class BlogCard extends StatefulWidget {

  final BlogPost post;

  const BlogCard({Key? key, required this.post}) : super(key: key);

  @override

  \_BlogCardState createState() => \_BlogCardState();

}

class \_BlogCardState extends State<BlogCard> {

  bool isHovered = false;

  @override

  Widget build(BuildContext context) {

    return MouseRegion(

      onEnter: (\_) => setState(() => isHovered = true),

      onExit: (\_) => setState(() => isHovered = false),

      child: AnimatedContainer(

        duration: const Duration(milliseconds: 200),

        decoration: BoxDecoration(

          color: Colors.grey[900],

          borderRadius: BorderRadius.circular(12),

          border: Border.all(

            color: isHovered ? Colors.purple : Colors.transparent,

            width: 2,

          ),

          boxShadow: [

            BoxShadow(

              color: isHovered ? Colors.purple.withOpacity(0.3) : Colors.black54,

              blurRadius: isHovered ? 8 : 4,

              offset: const Offset(0, 4),

            ),

          ],

        ),

        child: Column(

          crossAxisAlignment: CrossAxisAlignment.start,

          children: [

            // Blog Image

            ClipRRect(

              borderRadius: const BorderRadius.vertical(top: Radius.circular(12)),

              child: Image.network(

                widget.post.imageUrl,

                height: 150,

                width: double.infinity,

                fit: BoxFit.cover,

                errorBuilder: (context, error, stackTrace) => Container(

                  height: 150,

                  width: double.infinity,

                  color: Colors.grey[800],

                  child: const Icon(Icons.broken\_image, color: Colors.white),

                ),

              ),

            ),

            // Blog Text Content

            Padding(

              padding: const EdgeInsets.all(12.0),

              child: Column(

                crossAxisAlignment: CrossAxisAlignment.start,

                children: [

                  Text(

                    widget.post.title,

                    style: TextStyle(

                      fontSize: 20,

                      fontWeight: FontWeight.bold,

                      color: isHovered ? Colors.purple : Colors.white,

                    ),

                    maxLines: 1,

                    overflow: TextOverflow.ellipsis,

                  ),

                  const SizedBox(height: 6),

                  Text(

                    widget.post.summary,

                    style: TextStyle(

                      fontSize: 16,

                      color: Colors.grey[300],

                    ),

                    maxLines: 2,

                    overflow: TextOverflow.ellipsis,

                  ),

                  const SizedBox(height: 6),

                  Text(

                    widget.post.content,

                    style: TextStyle(

                      fontSize: 14,

                      color: isHovered ? Colors.purple[200] : Colors.grey[400],

                    ),

                    maxLines: 5,

                    overflow: TextOverflow.ellipsis,

                  ),

                  const SizedBox(height: 10),

                  // Read More Button → Show Dialog

                  Align(

                    alignment: Alignment.centerRight,

                    child: TextButton(

                      onPressed: () {

                        showDialog(

                          context: context,

                          builder: (BuildContext context) {

                            return AlertDialog(

                              backgroundColor: Colors.grey[900],

                              shape: RoundedRectangleBorder(

                                borderRadius: BorderRadius.circular(12),

                              ),

                              title: Text(

                                widget.post.title,

                                style: const TextStyle(color: Colors.white),

                              ),

                              content: SingleChildScrollView(

                                child: Text(

                                  widget.post.content,

                                  style: const TextStyle(color: Colors.white70),

                                ),

                              ),

                              actions: [

                                TextButton(

                                  child: const Text(

                                    "Close",

                                    style: TextStyle(color: Colors.purple),

                                  ),

                                  onPressed: () => Navigator.of(context).pop(),

                                ),

                              ],

                            );

                          },

                        );

                      },

                      style: TextButton.styleFrom(

                        foregroundColor: Colors.purple,

                      ),

                      child: const Text("Read More"),

                    ),

                  ),

                ],

              ),

            ),

          ],

        ),

      ),

    );

  }

}

class BlogScreen extends StatelessWidget {

final BlogPost post;

const BlogScreen({Key? key, required this.post}) : super(key: key);

static const List<BlogPost> blogPosts = [

BlogPost(

title: 'Dollar',

summary: 'The Reign of the Worlds Most Traded Currency',

content: '''In today’s hyperconnected financial world, one currency wields unmatched power and influence—the U.S. dollar. Often called the greenback, this iconic piece of American paper is more than just legal tender. It’s a symbol of global trust, a tool of international diplomacy, and a silent controller of economies. But how did the dollar achieve this dominance, and why does it continue to hold such a firm grip on the world?

The US Dollar isn’t just the currency of the United States—it’s the lifeblood of global finance. From oil deals in the Middle East to tech imports in Asia, the dollar rules international transactions. But how did it become so dominant, and what makes it stay on top? The dollar’s global dominance began after World War II with the Bretton Woods Agreement. This system pegged other major currencies to the dollar, which was itself backed by gold. Even after the gold standard ended in 1971, the dollar kept its power due to the strength of the US economy, political stability, and military and financial influence. Today, more than 60% of the world’s central bank reserves are held in dollars. Countries rely on the dollar to stabilize their economies and protect against currency risks. Its trust, liquidity, and worldwide acceptance make it the preferred reserve currency. The dollar is also the currency of global trade. Vital commodities like oil, gold, and gas are traded in dollars. This system, often referred to as the Petro-Dollar system, keeps global demand for the dollar high, even in transactions where the US isn’t involved. Beyond trade, the dollar is a powerful tool of influence. Because global payments often flow through US financial systems, the US can impose financial sanctions that block countries or companies from the dollar network. This makes it a non-military weapon used to isolate or punish regimes like Iran, Russia, or North Korea. The dollar is also a safe haven in uncertain times. During crises like wars, pandemics, or financial crashes, investors flock to the dollar. It’s seen as a stable store of value, and US Treasury bonds become highly sought after. This strengthens the dollar and gives the US financial advantages even during global turbulence. However, this dominance isn’t without consequences. Many developing countries borrow in US dollars. When the dollar strengthens, their debt becomes more expensive to repay, leading to inflation, slower growth, or financial crises in those nations. Even the US Federal Reserve’s decisions have global impact. When interest rates rise in the US, investors pull out of emerging markets, causing local currencies to weaken. The Fed’s policies ripple across continents, showing how central the dollar is to global finance. Lastly, the dollar’s dominance is self-reinforcing. Because everyone already uses it, it continues to be the default for trade, savings, and reserves. No other currency has yet matched its infrastructure, trust, and stability. While countries like China and Russia are working to reduce dependence on the dollar, and while cryptocurrencies are emerging, none currently offer the liquidity or trust needed to dethrone the greenback. For now, and perhaps for decades to come, the dollar remains king.

The Dollar as the Global Reserve Currency

One of the biggest reasons behind the dollar’s strength is its role as the global reserve currency. Around 60% of all central bank foreign reserves are held in U.S. dollars. Countries around the world trust the dollar to stabilize their own economies and protect against currency shocks.

This status began after World War II with the Bretton Woods Agreement, which pegged global currencies to the U.S. dollar, itself backed by gold at the time. Even after the gold standard was abandoned, the dollar retained its crown due to the sheer size and stability of the U.S. economy.

Dominating Global Trade

The dollar is the default currency for international trade, especially for vital resources like oil, gas, and gold. Whether it's a crude oil deal between China and Saudi Arabia or a tech purchase from Europe, chances are the transaction is settled in dollars.

This gives the U.S. a quiet yet powerful advantage—it becomes the middleman in most major trade deals, even when the U.S. isn’t directly involved. Other currencies simply don’t have the same level of trust, liquidity, or reach.

Sanctions: The Dollar as a Weapon

The dollar isn’t just a tool of trade—it’s also a weapon of control. Since most global transactions flow through U.S. banks or are cleared in dollars, America can use financial sanctions to cut off countries, companies, or individuals from the global economy.

Nations like Iran, North Korea, and Russia have felt the bite of this power. Being locked out of the dollar system can cripple economies and isolate regimes, giving the U.S. immense geopolitical leverage without firing a single shot.

A Global Safe Haven

During times of crisis—whether it's a war, pandemic, or market crash—investors flee to the safety of the dollar. It’s considered the world’s most reliable store of value. That’s why, in turbulent times, the dollar typically strengthens, and U.S. Treasury bonds become hot assets.

This global faith in the dollar allows the U.S. to borrow money at lower interest rates and maintain a strong economic position even when other countries are struggling.

Dollar-Denominated Debt: A Double-Edged Sword

Many countries and companies borrow in U.S. dollars due to its stability. But when the dollar strengthens, repaying that debt becomes more expensive. This creates financial stress in emerging markets, leading to inflation, defaults, or economic slowdowns.

Ironically, while the dollar gives global access to capital, it also becomes a burden when exchange rates shift, reinforcing American dominance and deepening dependence on the greenback.

The Fed’s Global Influence

The Federal Reserve—America’s central bank—doesn’t just set monetary policy for the U.S. It influences the entire world. When the Fed raises interest rates, capital flows out of developing countries and into American markets. This can trigger currency crashes and inflation overseas, showing how decisions made in Washington ripple across continents.

A Self-Reinforcing System

The dominance of the dollar is held up by a powerful network effect. Because everyone already uses it, new players prefer to stick with it, whether they’re building central bank reserves or conducting trade. This creates a cycle where the dollar remains on top, simply because it's already on top.''',

imageUrl: 'https://images.pexels.com/photos/164527/pexels-photo-164527.jpeg?auto=compress&cs=tinysrgb&w=400',

),

BlogPost(

title: 'Euro',

summary: 'Beyond Borders: The Euro and European Unity',

content:'''The Euro is not merely a currency — it is a revolutionary idea that reshaped the political, social, and economic fabric of Europe. Designed to unite diverse nations under a single monetary system, the Euro today stands as a global powerhouse, symbolizing strength, cooperation, and shared destiny.

But how did this ambitious project come into existence? Let's journey through the story of the Euro.

Historical Background: Seeds of Unity

After the catastrophic destruction of World War II, European nations realized that lasting peace could only be achieved through deep economic and political integration.

The first steps came with the formation of the European Coal and Steel Community (ECSC) in 1951, followed by the Treaty of Rome in 1957, which created the European Economic Community (EEC).

The idea was simple but powerful: nations that trade together and share economic interests are less likely to go to war.

Economic unity was seen as a foundation for political unity — and eventually, a shared currency would be the strongest symbol of that unity.

The Birth of the Euro

The dream of a single European currency became serious during the 1970s, but multiple financial crises delayed progress.

Finally, in 1992, the Maastricht Treaty formally established the blueprint for a common currency and economic convergence.

Key milestones:

1999: The Euro was introduced as a digital accounting currency for electronic payments and banking.

2002: Euro banknotes and coins were launched across 12 European countries, replacing their national currencies.

For the first time in history, sovereign nations voluntarily gave up their individual currencies in favor of a shared future.

The Euro thus became a unique experiment — never before had so many independent states agreed to use a single currency without a full political union.

Vision Behind the Euro

The creators of the Euro envisioned:

1) Economic Stability: Strengthen Europe's economy by eliminating currency fluctuations.

2) Deeper Political Bonds: Build trust and cooperation among member nations.

3) Global Influence: Position Europe as a major player against giants like the United States and China.

4) Convenience: Simplify cross-border trade, investment, and travel.

The Euro was meant to make Europe not just a collection of countries, but a true single market with seamless movement of goods, services, capital, and people.

The Eurozone: A Club of Shared Destiny

Today, the Eurozone includes 20 countries that have adopted the Euro as their official currency.

Together, they form a single monetary bloc governed by the European Central Bank (ECB).

The ECB controls monetary policy for the entire Eurozone, ensuring:

1 ) Inflation remains low and stable

2) Financial systems remain strong

3) Economic growth is sustainable

Countries like Germany, France, Italy, and Spain are major players, but smaller nations like Estonia and Slovenia also contribute to the richness of the Eurozone.

Advantages of the Euro

The Euro has unlocked numerous benefits:

1) Trade Boom: Cross-border business became faster, cheaper, and more predictable.

2) Travel Made Easy: No more changing currencies when traveling across most of Europe.

3)Price Transparency: Consumers can compare prices directly across countries.

4) Global Stability: The Euro provides a balance to the dominance of the U.S. dollar.

5) Political Integration: The currency acts as a daily reminder of a shared European identity.

Struggles and Criticism

Despite its successes, the Euro has faced serious hurdles:

Debt Crises: Countries like Greece, Portugal, and Ireland faced economic collapses in the late 2000s.

Diverging Economies: Wealthier countries like Germany have very different economic needs compared to countries like Greece or Spain.

Limited Flexibility: Member countries cannot print their own money or adjust exchange rates during a crisis.

Public Backlash: In some countries, the Euro was blamed for rising prices and economic hardship, fueling Euroscepticism.

The Euro’s Global Role

Today, the Euro is:

The second most used currency worldwide (after the U.S. dollar)

Widely used in international trade deals, central bank reserves, and financial markets

A symbol of Europe’s collective power in a globalized economy

Around 20% of global foreign exchange reserves are held in Euros, proving its strength and influence on the world stage.

The Euro and European Identity

More than economics, the Euro is a cultural and political bond:

People from different languages and traditions use the same currency daily.

The shared currency strengthens the sense of belonging to something larger than national borders.

Euro coins even celebrate national diversity, with one common side and one side unique to each country.

The Euro quietly weaves a story of unity into the daily lives of millions of Europeans.

The Road Ahead: Future of the Euro

The Euro's journey is still unfolding:

Digital Euro: Efforts are underway to launch a Central Bank Digital Currency (CBDC) version of the Euro.

Stronger Fiscal Unity: Discussions continue about deeper integration, including shared budgets and taxes.

Expansion: More EU countries are preparing to adopt the Euro in coming years.

Sustainability Focus: Green economic policies will play a central role in the Eurozone’s evolution.

The Euro will need to adapt to technological innovation, climate change, political shifts, and economic challenges to remain a powerful force in the 21st century.''',

imageUrl: 'https://images.pexels.com/photos/158776/euro-money-currency-the-european-158776.jpeg?auto=compress&cs=tinysrgb&w=400',

),

BlogPost(

title: 'Pound',

summary: 'Legacy Currency: The Pound’s Prestige & Power',

content: '''Birth of the Pound Sterling

The British Pound Sterling has an extraordinary heritage that dates back over 1,200 years. Its story begins during the reign of King Offa of Mercia in the 8th century.

Around 775 AD, Offa introduced the silver penny as a standard unit of currency across his kingdom. A Pound was originally defined as the weight of one pound of sterling silver, from which the currency took its name.

At that time, 240 silver pennies made up one Pound, establishing a monetary system that would influence British finance for centuries.

The term “Sterling” signified the high-quality silver used, and this reputation for purity and reliability set the foundation for Britain’s long-standing economic influence.

Growth During the Middle Ages: A Symbol of Wealth and Power

During the medieval period, the Pound became not just a means of trade, but a symbol of royal authority and national wealth.

Even though common people rarely held large sums, major transactions like land purchases, taxation, and debts were all calculated in Pounds, Shillings, and Pence.

The Royal Mint, established in London’s Tower of London, standardized coin production under the crown’s supervision, ensuring the Pound remained a trusted unit of value.

Over the centuries, monarchs like William the Conqueror and Henry II left their marks on England’s monetary system, while the Pound evolved into a recognized standard across Europe.

The Pound and the Rise of the British Empire

As Britain expanded into a global empire during the 16th to 19th centuries, the Pound Sterling traveled with explorers, merchants, and colonial administrators.

The currency became a vital tool in establishing Britain's dominance over international trade routes spanning North America, Asia, Africa, and Australia.

The founding of the Bank of England in 1694 marked a turning point, introducing paper banknotes backed by gold and silver reserves.

The Pound’s strength reflected Britain's political, naval, and commercial power, and London emerged as the world’s financial capital, with the Pound as the world's premier trading currency.

Golden Age: The Pound Under the Gold Standard

The 19th century ushered in the Pound’s golden age with Britain’s adoption of the Gold Standard in 1816.

This meant that every Pound note was backed by a fixed quantity of gold, giving international traders and governments immense confidence in the British currency.

London’s financial markets thrived, and the Pound Sterling became the world's dominant reserve currency.

For much of the 1800s, Britain’s industrial revolution and colonial expansion ensured that Sterling was the backbone of the global economy, rivaled by none.

War, Crisis, and Decline: Sterling Faces Challenges

The outbreak of World War I in 1914 marked the beginning of significant challenges for the Pound.

Britain had to suspend the Gold Standard to finance its war efforts, causing inflation and economic instability.

Although the country attempted to return to the Gold Standard briefly in 1925, the Great Depression forced its permanent abandonment in 1931.

World War II further strained Britain's economy, and by the late 1940s, the U.S. Dollar replaced the Pound as the primary global reserve currency.

Despite losing its preeminent status, the Pound continued to symbolize British resilience through economic reconstruction and social change.

Decimalisation: Modernizing the Pound

For centuries, Britain’s monetary system was notoriously complex, with 240 pence in one Pound and 12 pence in a shilling.

On February 15, 1971, famously known as "Decimal Day," Britain reformed its currency system to make it simpler:

1 Pound = 100 new pence.

This modernization made the Pound easier to use both domestically and internationally, aligning the UK’s financial system with the rest of the modern world.

The Euro Debate: Choosing Sovereignty Over Integration

In the late 1990s, as the European Union introduced the Euro, Britain faced immense political debate over whether to adopt the new currency.

Despite pressures from businesses and some politicians, public opinion strongly favored keeping the Pound.

Many British citizens viewed the Pound not just as a currency, but as a symbol of national identity, heritage, and sovereignty.

Ultimately, Britain opted out of the Eurozone, maintaining the Pound Sterling and reinforcing its importance to British national pride.

Unique Features of the Modern Pound

Today’s Pound notes and coins are rich in symbolism and history.

New polymer banknotes feature detailed portraits of British icons such as Winston Churchill, Jane Austen, and Alan Turing.

Coins feature emblems representing the four nations of the United Kingdom: the English Rose, the Scottish Thistle, the Welsh Leek, and the Northern Irish Shamrock.

Furthermore, Scotland and Northern Ireland issue their own unique designs of Pound banknotes, reflecting regional identities within the UK.

Timeline: Key Milestones of the British Pound

775 AD — King Offa introduces the silver penny.

1158 — King Henry II standardizes the English currency system.

1694 — Bank of England founded; first Pound banknotes issued.

1816 — Britain formally adopts the Gold Standard.

1931 — Permanent exit from the Gold Standard.

1971 — Decimalisation simplifies the Pound system.

1999 — Britain decides against joining the Euro.

The Enduring Power of the Pound Sterling

The story of the British Pound Sterling is the story of Britain itself — a journey through kingdoms, empires, wars, revolutions, and globalization.

Despite facing immense global challenges over the centuries, the Pound remains a resilient and highly respected currency, trusted both at home and abroad.

In an era where many currencies have disappeared or been replaced, the Pound Sterling continues to stand tall, reflecting over a millennium of history, strength, and tradition.

To this day, holding a Pound note is like holding a tangible piece of history — a symbol of a nation that shaped the world.

''',

imageUrl: 'https://images.pexels.com/photos/1791583/pexels-photo-1791583.jpeg?auto=compress&cs=tinysrgb&w=400',

),

BlogPost(

title: 'Rupee',

summary: 'Rising Rupee: India’s Growing Global Footprint',

content: '''The Rupee has one of the longest and richest histories among world currencies.

Its roots trace back to the 16th century, during the rule of Sher Shah Suri, an Afghan ruler who controlled much of northern India.

In 1540, Sher Shah introduced a standardized silver coin called the "Rupiya," weighing around 178 grains (approximately 11.53 grams).

The term “Rupiya” comes from the Sanskrit word “Rupyakam,” meaning "wrought silver" or "a silver coin."

Before the Rupiya, various regions in India used different forms of metal-based money, but Sher Shah’s reform unified the system, making trade and governance much more efficient across the Indian subcontinent.

His innovation laid the foundation for India's monetary tradition, which survived multiple dynasties, colonizers, and modern economic transformations.

Evolution Through Mughal, Maratha, and Colonial Periods

Following Sher Shah, the Mughal emperors continued using the Rupiya.

Akbar the Great (1556–1605), a visionary ruler, established a well-organized financial system based on the silver Rupiya.

Under the Mughal Empire, the coinage was highly respected for its purity and uniformity, helping India's trade flourish with Persia, Arabia, and even parts of Europe.

Later, during the Maratha Confederacy and various princely states' rule, local variations of the Rupiya appeared, yet the core idea of a silver-based currency stayed intact.

With the rise of British East India Company in the 18th century, the Rupee was absorbed into the colonial system.

The British gradually shifted coin production to mints they controlled and standardized the design of Rupees under Queen Victoria’s authority.

By the 19th century, the Indian Rupee had become a critical part of global trade networks across Asia and Africa, under British imperial influence.

Transition from Silver Standard to Paper Currency

Originally, the Rupee was strictly linked to silver, unlike the British Pound, which was gold-backed.

As the global price of silver fell in the late 19th century due to massive discoveries of silver mines in the Americas, India’s silver-backed Rupee lost value compared to gold-backed currencies.

To manage this crisis, the British colonial administration introduced Paper Rupees in 1862 — marked by the Queen's portrait and issued by the Government of India.

This marked a critical shift — from a purely metallic standard to a modern monetary system based on the faith and stability of government-issued paper money.

Banks were established, and the use of the Rupee in large commercial transactions became widespread.

Despite colonial exploitation, this period laid crucial groundwork for India's later economic development.

A Currency of Sovereignty

With India's independence in 1947, the Rupee underwent not just a visual makeover but also a philosophical one.

Gone were the symbols of colonial power.

They were replaced with national symbols like the Lion Capital of Ashoka, portraits of Mahatma Gandhi, and images showcasing India's rich cultural and technological heritage.

The first post-independence note — a 1 Rupee note — still carried King George VI’s image briefly before it was replaced in 1949 with completely Indian motifs.

The Indian government established the Reserve Bank of India (RBI) as the sole issuer of currency, giving the Rupee its modern-day governance structure.

Thus, the Rupee became a symbol not just of commerce, but also of India’s pride, self-reliance, and national unity.

Major Events: Devaluation, Economic Liberalization, and Globalization

The Rupee’s journey was not without hardship.

In 1966, under pressure from the International Monetary Fund (IMF) and due to a growing balance of payments crisis, India devalued the Rupee drastically against the US Dollar.

This move was highly controversial and reflected India's struggle to balance its socialist economic model with global trade realities.

Again in the 1990s, facing a severe financial crisis, India undertook sweeping economic reforms under the leadership of Dr. Manmohan Singh, then Finance Minister.

The liberalization policies of 1991 opened India’s markets to foreign investment, drastically changing the Rupee’s role.

The Rupee became convertible on the current account in 1994, meaning it could be freely exchanged for foreign currencies for trade purposes — a massive leap towards globalization.

Modern Era: Technology, Security, and the Rise of the Digital Rupee

Fast forward to today, the Rupee is adapting to new challenges posed by technology and global finance.

In 2010, the Indian government introduced a unique currency symbol (₹), blending Indian and Latin script elements — a move to brand the Rupee internationally, similar to the Dollar and Euro .

In 2016, India announced a sudden demonetization of the 500 and 1000 Rupee notes to curb black money and corruption, reshaping the public’s interaction with physical currency.

New notes with improved security features — including color-changing ink, micro-lettering, and see-through registration marks — were introduced.

Now, with the Digital India mission, the Digital Rupee (CBDC) project is underway, preparing India to launch a central bank digital currency that could revolutionize banking, reduce transaction costs, and make financial systems more efficient.

The Global Presence and Symbolism of the Rupee

The Indian Rupee is widely accepted in neighboring countries like Nepal and Bhutan.

It holds a unique status among global currencies, not just because of India's population size, but also due to its fast-growing economy and influence in global affairs.

Moreover, the Rupee's design — with 15 official languages printed on every note — reflects India’s diverse and inclusive culture.

From intricate architectural designs on banknotes to the portrait of Mahatma Gandhi, the Rupee tells the world a story of unity in diversity, resilience, and progress.

Timeline of Major Milestones

1540 — Rupiya introduced by Sher Shah Suri.

1600s-1800s — Mughal and British versions of the Rupiya dominate.

1862 — Introduction of Paper Rupees under British India.

1947 — Indian Independence; new national Rupee begins.

1966 — Major devaluation of the Rupee.

1991 — Economic liberalization begins.

2010 — Introduction of the ₹ symbol.

2016 — Demonetization and introduction of the new series.

2023-24 — Pilot launch of India's Digital Rupee (CBDC).

Interesting Facts About the Rupee

The first Indian banknote (₹10) was printed under the British in 1861.

Nepal and Bhutan accept the Indian Rupee at par value in their countries.

Each Indian banknote shows 15 languages besides Hindi and English.

The highest denomination note in Indian history was ₹10,000 (now demonetized).

The Rupee — More Than Just Money

The Rupee is not merely a tool for transactions; it is a symbol of India’s heritage, a witness to its struggles and triumphs, and a beacon of its future aspirations.

From the silver coins that ruled ancient bazaars to the secure, digitally-enabled currency shaping a trillion-dollar economy today, the journey of the Rupee mirrors the story of India itself — dynamic, resilient, and forward-looking.

As India continues its journey toward becoming one of the world's leading economies, the Rupee will remain a pillar of national pride, standing testament to centuries of economic, political, and cultural evolution.''',

imageUrl: 'data:image/jpeg;base64,',

),

BlogPost(

title: 'Dirham',

summary: 'Desert Gold: The Rise of the Emirati Dirham',

content: '''The history of the Dirham is deeply rooted in the rich exchanges of ancient civilizations.

The term "Dirham" traces its origins to the Greek coin 'Drachma', which was widely circulated throughout the Mediterranean world.

As trade routes expanded from Europe into the Middle East during the classical era, the concept of coinage — and the name itself — traveled along with merchants, influencing early Arab economies.

By the 7th century, during the rise of the Islamic Caliphate, the Dirham had been fully adopted by the Muslim world.

Caliphates minted silver Dirhams that carried religious inscriptions and became crucial instruments of commerce.

The Dirham thus evolved from a Greek idea into an Islamic symbol of economic and cultural unity across a vast empire stretching from Spain to Central Asia.

Dirham in the Islamic Golden Age: Economic Backbone of a Civilization

The Dirham played a pivotal role during the Islamic Golden Age (8th–13th centuries), a time when Muslim societies led the world in trade, science, medicine, and the arts.

Markets across Baghdad, Cairo, Damascus, and Cordoba depended on the Dirham for transactions large and small.

It wasn't just a medium of exchange — it became a trusted symbol of value for scholars, travelers, and merchants alike.

Inscriptions on Dirhams were often minimalist, focusing on Quranic verses rather than rulers' images, which reflected Islamic values.

These coins were crucial to financing monumental projects like libraries, universities, and sprawling trade networks linking Europe, Africa, and Asia.

The Dirham's Role in International Trade Networks

The influence of the Dirham wasn’t confined to Islamic territories alone.

Due to the expansive trading networks, Dirham coins have been found in places as far-flung as Russia, Scandinavia, and China.

Viking explorers, for example, prized silver Dirhams and often used them as raw material for jewelry and trade.

The Dirham's acceptance on the Silk Road and throughout the Indian Ocean trade routes showcases its incredible importance in connecting civilizations.

It became a unifying financial instrument in a time when no centralized global economy existed.

Colonial Influences and Shifting Currencies in the Gulf

Fast forward several centuries: by the 19th and early 20th centuries, the Arabian Peninsula was experiencing massive changes.

European colonial powers, especially the British, dominated the region’s trade and political structures.

As a result, currencies like the Indian Rupee and the Gulf Rupee (a special version issued for Gulf countries) replaced traditional coinage systems, including the Dirham.

The indigenous currencies faded as the Gulf economies became more dependent on colonial trade networks.

However, the memory and legacy of the Dirham remained alive, symbolizing a proud heritage that awaited revival.

Revival of the Dirham: Formation of the Modern UAE Currency

The opportunity to revive a native currency came after the formation of the United Arab Emirates on December 2, 1971.

After a brief period of using the Qatari and Dubai Riyal, the UAE officially introduced the United Arab Emirates Dirham (AED) on May 19, 1973.

This marked a significant moment — not just an economic move, but a cultural assertion of independence, unity, and national pride.

The UAE Central Bank took charge of issuing and regulating the Dirham, designing it to reflect the country's rich history, traditions, and modern aspirations.

Design and Features of the UAE Dirham

The design of the Dirham reflects both tradition and progress:

Coins depict cultural symbols like the Dallah (traditional coffee pot), oil rigs, and dhows (traditional sailing vessels).

Banknotes feature stunning illustrations of UAE landmarks: mosques, forts, and modern skyscrapers.

All Dirhams have inscriptions in Arabic on the front and English on the back, symbolizing the UAE’s global openness.

Modern Dirhams have advanced security features, including watermarks, holographic strips, color-shifting ink, and micro-printing.

Dirham coins and notes are available in multiple denominations, serving both daily needs and large transactions in one of the world’s fastest-growing economies.

Economic Stability: Pegged to the US Dollar

Since 1997, the Dirham has been pegged to the US Dollar at a fixed rate of 1 USD = 3.6725 AED.

This peg brings remarkable financial stability to the UAE’s economy, boosting investor confidence and enabling predictable international trade.

The Dirham’s fixed value supports major sectors such as oil exports, real estate, aviation, and global tourism, establishing the UAE as a premier business hub.

The Dirham’s Global Impact

Today, the Dirham is widely recognized and respected around the world.

From luxury shopping in Dubai’s malls to international real estate deals, the Dirham facilitates billions of dollars in global transactions each year.

Moreover, millions of expatriates working in the UAE remit billions of Dirhams back home annually, making it one of the top remitted currencies worldwide.

Whether in a bustling souk or in a high-tech financial center, the Dirham remains at the heart of the UAE’s economic miracle.

Timeline Highlights

6th century BC — Greek Drachma emerges.

7th century AD — Islamic Dirham introduced.

8th-13th centuries — Dirham flourishes during Islamic Golden Age.

19th-20th centuries — Gulf Rupee replaces local coinage.

1971 — UAE founded.

1973 — Modern UAE Dirham launched.

1997 — Dirham officially pegged to US Dollar.

Fascinating Facts About the Dirham

Ancient Dirhams were so trusted that even Viking warriors hoarded them in their treasure troves.

The name "Dirham" survived for centuries even during periods when it wasn't officially used.

Modern Dirham coins are made from a cupronickel alloy, making them resistant to corrosion from the Gulf's humid climate.

The UAE Dirham features no images of political leaders, respecting Islamic traditions about human depictions.

A Currency That Embodies Legacy and Ambition

The Dirham is much more than a monetary tool.

It represents a rich tapestry of civilizations, a journey through ancient empires, colonial struggles, and the bold vision of a modern nation.

As the UAE continues to lead in innovation, space exploration, and global finance, the Dirham remains a proud emblem of resilience, unity, and boundless ambition.

It carries within it not only the history of trade and empire but also the dreams of a future yet to be written.''',

imageUrl: 'https://media.istockphoto.com/id/1211295986/photo/united-arab-emirates-money-a-business-background.jpg?b=1&s=612x612&w=0&k=20&c=VoIGTbGvHGo3K0sUZhp3BT8GZEu\_HB-ZwbL6g7b3--w=',

),

BlogPost(

title: 'Dinar',

summary: 'Oil and the Dinar: Wealth, War, and Worth',

content: '''A Symbol of Wealth and Prestige

The Dinar is more than just a currency; it represents centuries of wealth, authority, and cultural pride.

From ancient Islamic caliphates to modern oil-rich nations, the Dinar has maintained a prestigious place in the world economy.

It stands today as a living reminder of the golden ages of civilization, trade, and financial innovation.

The Historical Genesis of the Dinar: From Rome to Islam

The story of the Dinar begins with the Roman Denarius, a silver coin circulated across the Roman Empire.

As Islam spread across Arabia and beyond, the need for an independent and dignified Islamic currency arose.

In 696 AD, under the leadership of Caliph Abd al-Malik ibn Marwan, the first pure gold Islamic Dinar was minted.

This marked a revolutionary moment — a currency not bearing images of emperors, but instead inscribed with Quranic verses and Islamic declarations, emphasizing faith and sovereignty.

Key Highlights of the Early Dinar:

1) Weight: ~4.25 grams of pure gold

2) Circular shape with Arabic inscriptions

No images of humans or animals (reflecting Islamic principles)

This Islamic Dinar set a new standard of purity, faith, and economy.

How the Dinar Fueled the Expansion of Islamic Trade Networks

The Dinar was not just a local currency — it was the backbone of a massive trading empire.

During the Islamic Golden Age, merchants, travelers, and scholars used Dinars to facilitate trade across:

The Silk Road (linking China to the Middle East)

The Indian Ocean routes (connecting India, East Africa, and Arabia)

The Trans-Saharan routes (trading gold, salt, and goods across Africa)

Its purity made it trusted beyond Muslim lands — even Viking merchants in Scandinavia and traders in China accepted the Dinar as a reliable medium of exchange.

The Dinar's Deep Religious and Political Importance

In Islamic tradition, wealth had to meet ethical and moral standards — and the Dinar fit perfectly.

Religious Significance:

The Dinar was often mentioned in the context of Zakat (charity obligations).

It symbolized halal earnings — clean, pure, and just wealth.

Political Symbolism:

Minting the Dinar showed sovereign independence from foreign empires.

It reflected a ruler’s legitimacy and control over economic matters.

Every Dinar carried not just financial value but religious, political, and cultural pride.

The Global Footprint of the Dinar in Ancient Economies

Due to its consistency and high gold content, the Dinar spread far and wide:

1) Found buried in Viking hoards in Northern Europe.

2) Used in Byzantine markets alongside other coins.

3) Circulated through Chinese bazaars via the Silk Road.

The Dinar became a true global currency of its time — centuries before modern globalization.

Challenges and Decline: Why the Classical Dinar Faded

Despite its dominance, the Dinar’s influence declined due to:

1) Political fragmentation of Islamic empires

2) Mongol invasions disrupting trade routes

Rise of European colonial currencies (like the Spanish Real and British Pound)

By the late Middle Ages, silver and other base metals began replacing gold coins in many economies.

Still, the Dinar’s legacy remained engraved in history and memory.

🇯🇴 Modern Resurrection: Nations that Embrace the Dinar Today

Several countries proudly carry the Dinar legacy:

Country Currency Name Interesting Fact

1) Jordan Jordanian Dinar (JOD) Pegged strongly to the US dollar

2) Kuwait Kuwaiti Dinar (KWD) World's most valuable currency

3) Iraq Iraqi Dinar (IQD) Rebuilding post-conflict economy

4) Bahrain Bahraini Dinar (BHD) Oil-backed strong currency

5) Algeria Algerian Dinar (DZD) Introduced after independence

6) Libya Libyan Dinar (LYD) Stabilized after political turmoil

Tunisia Tunisian Dinar (TND) Symbol of post-colonial freedom

Each modern Dinar represents not just economic policy but national identity and pride.

The Kuwaiti Dinar: World's Strongest Currency

Among all Dinars, the Kuwaiti Dinar (KWD) stands at the top:

.) 1 KWD ≈ 3.25 USD (highest exchange rate globally)

.) Supported by vast oil reserves

.) Strong, stable monetary policies

.) Limited inflation compared to other economies

The Kuwaiti Dinar shows how strong governance can make ancient legacies thrive in the modern world.

Art and Culture on Dinar Banknotes and Coins

Modern Dinar banknotes and coins are more than just money — they are cultural artifacts.

Common design features:

1) Historic monuments (like Petra, ancient cities)

2) Calligraphy and Islamic art

3) Portraits of national heroes and kings

4) Native wildlife and flora

Through these designs, every Dinar connects citizens to their history and culture daily.

The Economic Power of the Dinar in Today's World

.) Today, the Dinar plays a crucial role in:

1) Oil trade agreements (Kuwait, Iraq, Bahrain)

2)Tourism sectors (Tunisia, Jordan)

3) Regional investments and banking

Despite facing challenges like regional conflicts and economic pressures, Dinar economies have shown remarkable resilience, adapting to global market changes while preserving their historical identity.

.) Interesting Facts About the Dinar You Might Not Know

.) The first Islamic Dinar had no images — only Quranic text.

.) Kuwaiti Dinar is over 3x stronger than USD.

Ancient Dinars are now sold at auctions for thousands of dollars.

Some Islamic scholars today advocate for the return of the Gold Dinar in modern Islamic banking.

Dinar — A Timeless Legacy of Faith, Trade, and Power

The Dinar is a living story of the Islamic world's glorious past and hopeful future.

From the gold coins of Damascus to the shining banks of Kuwait City, the Dinar symbolizes resilience, pride, sovereignty, and excellence.

It remains a beacon of cultural identity and financial strength — a testament that great legacies never fade, they evolve .''',

imageUrl: 'https://images.pexels.com/photos/10920552/pexels-photo-10920552.jpeg?auto=compress&cs=tinysrgb&w=400',

),

];

get subtitleFontSize => null;

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: const Color.fromARGB(255, 45, 20, 55),

drawer: \_buildDrawer(context),

appBar: AppBar(

backgroundColor: const Color.fromARGB(255, 18, 6, 29),

iconTheme: const IconThemeData(color: Colors.white),

title: Row(

children: [

Image.asset('assets/logo.png', height: 40),

const SizedBox(width: 10),

const Text(

'Currency App',

style: TextStyle(color: Colors.white),

),

// Expanded(

// child: Text(

// post.title,

// style: const TextStyle(color: Colors.white, fontSize: 18),

// overflow: TextOverflow.ellipsis,

// ),

// ),

],

),

actions: [

IconButton(

icon: const Icon(Icons.account\_circle, color: Colors.white),

onPressed: () {

final user = FirebaseAuth.instance.currentUser;

if (user != null) {

\_showProfileDialog(context, user); // Show profile if logged in

} else {

Navigator.pushNamed(context, EmailPasswordLogin.routeName); // Go to login

}

},

),

PopupMenuButton<String>(

icon: const Icon(Icons.more\_vert, color: Colors.white),

onSelected: (value) async {

if (value == 'signout') {

await context.read<FirebaseAuthMethods>().signOut(context);

Navigator.pushAndRemoveUntil(

context,

MaterialPageRoute(builder: (\_) => const EmailPasswordLogin()),

(route) => false,

);

} else if (value == 'delete') {

await context.read<FirebaseAuthMethods>().deleteAccount(context);

} else if (value == 'signup') {

Navigator.pushNamed(context, EmailPasswordSignup.routeName);

} else if (value == 'login') {

Navigator.pushNamed(context, EmailPasswordLogin.routeName);

}

},

itemBuilder: (context) {

final user = FirebaseAuth.instance.currentUser;

if (user == null) {

return [

const PopupMenuItem(value: 'signup', child: Text('Sign Up')),

const PopupMenuItem(value: 'login', child: Text('Login')),

];

} else {

return const [

PopupMenuItem(value: 'signout', child: Text('Sign Out')),

PopupMenuItem(value: 'delete', child: Text('Delete Account')),

];

}

},

),

],

),

body: LayoutBuilder(

builder: (context, constraints) {

int crossAxisCount;

if (constraints.maxWidth > 1200) {

crossAxisCount = 4; // For very wide desktops

} else if (constraints.maxWidth > 900) {

crossAxisCount = 3; // For standard desktops/large tablets

} else if (constraints.maxWidth > 600) {

crossAxisCount = 2; // For smaller tablets/larger phones

} else {

crossAxisCount = 1; // For phones

}

final horizontalPadding = constraints.maxWidth \* 0.05;

final verticalPadding = constraints.maxWidth \* 0.02;

late double titleFontSize;

late double summaryFontSize;

late double contentFontSize;

late double cardHorizontalPadding;

late double cardVerticalPadding;

late double spacingBetween;

double cardAspectRatio = 0.7;

if (constraints.maxWidth > 1200) {

crossAxisCount = 4;

titleFontSize = constraints.maxWidth \* 0.017;

summaryFontSize = constraints.maxWidth \* 0.012;

contentFontSize = constraints.maxWidth \* 0.019;

cardHorizontalPadding = constraints.maxWidth \* 0.01;

cardVerticalPadding = constraints.maxWidth \* 0.01;

spacingBetween = constraints.maxWidth \* 0.01;

cardAspectRatio = 0.8; // Changed from 0.8 to 0.9

} else if (constraints.maxWidth > 900) {

crossAxisCount = 3;

titleFontSize = constraints.maxWidth \* 0.02;

summaryFontSize = constraints.maxWidth \* 0.015;

contentFontSize = constraints.maxWidth \* 0.014;

cardHorizontalPadding = constraints.maxWidth \* 0.01;

cardVerticalPadding = constraints.maxWidth \* 0.01;

spacingBetween = constraints.maxWidth \* 0.01;

cardAspectRatio = 1.2; // Changed from 1.1 to 1.2

} else if (constraints.maxWidth > 600) {

crossAxisCount = 2;

titleFontSize = constraints.maxWidth \* 0.025;

summaryFontSize = constraints.maxWidth \* 0.02;

contentFontSize = constraints.maxWidth \* 0.015;

cardHorizontalPadding = constraints.maxWidth \* 0.02;

cardVerticalPadding = constraints.maxWidth \* 0.015;

spacingBetween = constraints.maxWidth \* 0.01;

cardAspectRatio = 0.7;

} else {

crossAxisCount = 1;

titleFontSize = constraints.maxWidth \* 0.03;

summaryFontSize = constraints.maxWidth \* 0.025;

contentFontSize = constraints.maxWidth \* 0.02;

cardHorizontalPadding = constraints.maxWidth \* 0.02;

cardVerticalPadding = constraints.maxWidth \* 0.015;

spacingBetween = constraints.maxWidth \* 0.01;

cardAspectRatio = 0.7;

}

// Add these breakpoints - ensuring crossAxisCount is also set here

if (constraints.maxWidth > 1200 && constraints.maxWidth < 1400) {

crossAxisCount = 3;

titleFontSize = constraints.maxWidth \* 0.018;

summaryFontSize = constraints.maxWidth \* 0.013;

contentFontSize = constraints.maxWidth \* 0.012;

cardAspectRatio = 1.1; // Changed from 1.0 to 1.1

}

if (constraints.maxWidth > 900 && constraints.maxWidth < 1028) {

crossAxisCount = 2;

titleFontSize = constraints.maxWidth \* 0.023;

summaryFontSize = constraints.maxWidth \* 0.018;

contentFontSize = constraints.maxWidth \* 0.015;

cardAspectRatio = 1.1; // Changed from 1.0 to 1.1

}

if (constraints.maxWidth > 600 && constraints.maxWidth < 684) {

crossAxisCount = 1;

titleFontSize = constraints.maxWidth \* 0.028;

summaryFontSize = constraints.maxWidth \* 0.023;

contentFontSize = constraints.maxWidth \* 0.018;

cardAspectRatio = 0.7;

}

return Padding(

padding: EdgeInsets.symmetric(horizontal: horizontalPadding, vertical: verticalPadding),

child: SingleChildScrollView(

child: Column(

children: [

Text(

'Latest Blogs',

style: TextStyle(

fontSize: titleFontSize,

fontWeight: FontWeight.bold,

color: Colors.white,

),

),

const SizedBox(height: 8),

Text(

'Explore the world of currencies with insightful articles.\nStay updated with trends and news.',

style: TextStyle(

fontSize: subtitleFontSize,

fontWeight: FontWeight.w500,

color: Colors.white70,

),

textAlign: TextAlign.center,

),

const SizedBox(height: 20),

GridView.builder(

shrinkWrap: true,

physics: const NeverScrollableScrollPhysics(),

itemCount: blogPosts.length,

gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(

crossAxisCount: crossAxisCount,

crossAxisSpacing: 16,

mainAxisSpacing: 16,

childAspectRatio: 0.7,

),

itemBuilder: (context, index) {

return BlogCard(post: blogPosts[index]);

},

),

const SizedBox(height: 20),

const Divider(thickness: 1, color: Color.fromARGB(59, 243, 239, 239)),

const CustomFooter(),

],

),

),

);

},

),

);

}

void \_showProfileDialog(BuildContext context, User user) {

showDialog(

context: context,

builder: (\_) {

return AlertDialog(

title: const Text('Profile Details'),

content: Column(

mainAxisSize: MainAxisSize.min,

children: [

if (user.email != null) Text('Email: ${user.email}'),

if (!user.emailVerified)

ElevatedButton(

onPressed: () {

context

.read<FirebaseAuthMethods>()

.sendEmailVerification(context);

},

child: const Text('Verify Email'),

),

],

),

actions: [

TextButton(

onPressed: () => Navigator.pop(context),

child: const Text('Close'),

),

],

);

},

);

}

Widget \_buildDrawer(BuildContext context) {

final user = FirebaseAuth.instance.currentUser;

return Drawer(

backgroundColor: const Color.fromARGB(255, 18, 6, 29),

child: ListView(

padding: EdgeInsets.zero,

children: [

const DrawerHeader(

decoration: BoxDecoration(

gradient: LinearGradient(

colors: [

Color.fromARGB(255, 44, 1, 49),

Color.fromARGB(99, 89, 2, 97),

],

),

),

child: Text('Menu',

style: TextStyle(color: Colors.white, fontSize: 24)),

),

ListTile(

leading: const Icon(Icons.home, color: Colors.white),

title: const Text('Home', style: TextStyle(color: Colors.white)),

onTap: () {

Navigator.pop(context);

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (\_) => const home.HomeScreen()),

);

},

),

ExpansionTile(

leading: const Icon(Icons.settings, color: Colors.white),

title:

const Text('Settings', style: TextStyle(color: Colors.white)),

iconColor: Colors.white,

collapsedIconColor: Colors.white,

children: [

\_buildDrawerItem(

context: context,

icon: Icons.notifications\_active,

text: 'Rate Alerts',

screen: const RateAlertsScreen(),

),

\_buildDrawerItem(

context: context,

icon: Icons.trending\_up,

text: 'Currency News',

screen: const CurrencyNewsScreen(),

),

\_buildDrawerItem(

context: context,

icon: Icons.support\_agent,

text: 'Help Center',

screen: const HelpCenterScreen(),

),

if (user != null)

\_buildDrawerItem(

context: context,

icon: Icons.history,

text: 'Conversion History',

screen: const ConversionHistoryScreen(),

),

if (user != null)

\_buildDrawerItem(

context: context,

icon: Icons.currency\_exchange,

text: 'Default Currency',

screen: const DefaultCurrencyScreen(),

),

],

),

ListTile(

leading: const Icon(Icons.info, color: Colors.white),

title: const Text('Blog', style: TextStyle(color: Colors.white)),

onTap: () {

Navigator.pop(context);

Navigator.pushReplacement(

context,

MaterialPageRoute(

builder: (\_) => BlogScreen(

post: BlogScreen.blogPosts[0],

),

),

);

},

),

ListTile(

leading: const Icon(Icons.star, color: Colors.white),

title: const Text('Testimonial',

style: TextStyle(color: Colors.white)),

onTap: () {

Navigator.pop(context);

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (\_) => const TestimonialPage()),

);

},

),

ListTile(

leading: const Icon(Icons.call, color: Colors.white),

title: const Text('Contact', style: TextStyle(color: Colors.white)),

onTap: () {

Navigator.pop(context);

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (\_) => const ContactPage()),

);

},

),

],

),

);

}

Widget \_buildDrawerItem({

required BuildContext context,

required IconData icon,

required String text,

required Widget screen,

}) {

return ListTile(

leading: Icon(icon, color: Colors.white),

title: Text(text, style: const TextStyle(color: Colors.white)),

onTap: () {

Navigator.pop(context);

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (\_) => screen),

);

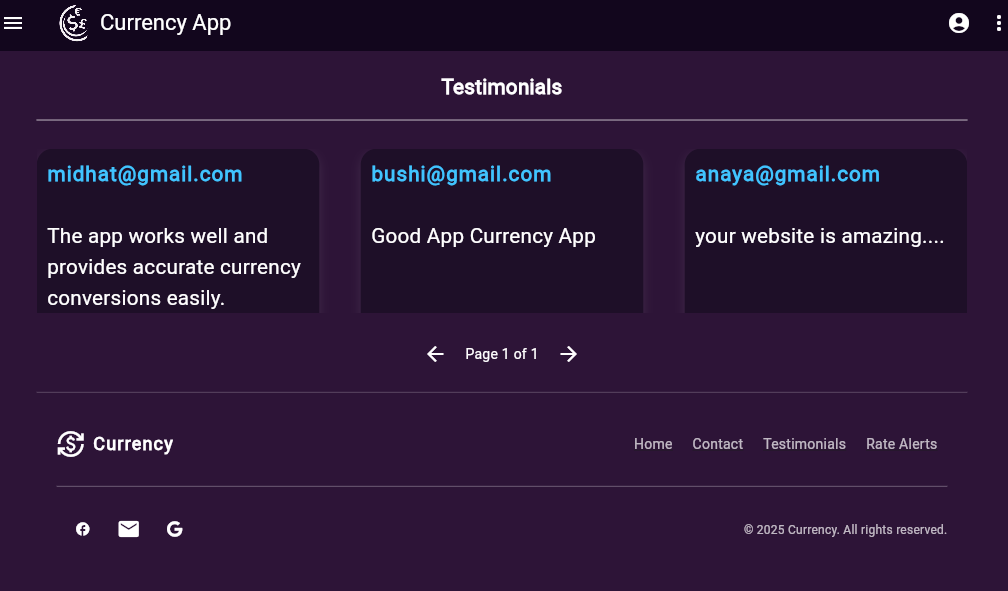
},

);

}

}

**Testimonial**

****

class TestimonialPage extends StatefulWidget {

  const TestimonialPage({Key? key}) : super(key: key);

  @override

  State<TestimonialPage> createState() => \_TestimonialPageState();

}

class \_TestimonialPageState extends State<TestimonialPage> {

  int \_currentPage = 0;

  final int \_itemsPerPage = 3;

  @override

  Widget build(BuildContext context) {

    final user = FirebaseAuth.instance.currentUser;

    return Scaffold(

      backgroundColor: const Color.fromARGB(255, 45, 20, 55),

      appBar: AppBar(

        backgroundColor: const Color.fromARGB(255, 18, 6, 29),

        iconTheme: const IconThemeData(color: Colors.white),

        title: Row(

          children: [

            Image.asset('assets/logo.png', height: 40),

            const SizedBox(width: 10),

            const Text(

              'Currency App',

              style: TextStyle(color: Colors.white),

            ),

          ],

        ),

        actions: [

          IconButton(

            icon: const Icon(Icons.account\_circle, color: Colors.white),

            onPressed: () {

              final user = FirebaseAuth.instance.currentUser;

              if (user != null) {

                \_showProfileDialog(context, user); // Show profile if logged in

              } else {

                Navigator.pushNamed(

                    context,

                    EmailPasswordLogin

                        .routeName); // Show login screen if not logged in

              }

            },

          ),

          PopupMenuButton<String>(

            icon: const Icon(Icons.more\_vert, color: Colors.white),

            onSelected: (value) async {

              if (value == 'signout') {

                await context.read<FirebaseAuthMethods>().signOut(context);

                Navigator.pushAndRemoveUntil(

                  context,

                  MaterialPageRoute(builder: (\_) => const EmailPasswordLogin()),

                  (route) => false,

                );

              } else if (value == 'delete') {

                await context

                    .read<FirebaseAuthMethods>()

                    .deleteAccount(context);

              } else if (value == 'signup') {

                Navigator.pushNamed(context,

                    EmailPasswordSignup.routeName); // Navigate to signup screen

              } else if (value == 'login') {

                Navigator.pushNamed(context,

                    EmailPasswordLogin.routeName); // Navigate to login screen

              }

            },

            itemBuilder: (context) {

              final user = FirebaseAuth.instance.currentUser;

              // If the user is not logged in, show login/signup options

              if (user == null) {

                return [

                  const PopupMenuItem(value: 'signup', child: Text('Sign Up')),

                  const PopupMenuItem(value: 'login', child: Text('Login')),

                ];

              } else {

                // If the user is logged in, show sign out/delete options

                return const [

                  PopupMenuItem(value: 'signout', child: Text('Sign Out')),

                  PopupMenuItem(value: 'delete', child: Text('Delete Account')),

                ];

              }

            },

          ),

        ],

      ),

      drawer: \_buildDrawer(),

      body: LayoutBuilder(

        builder: (context, constraints) {

          // Determine crossAxisCount based on available width

          int crossAxisCount;

          if (constraints.maxWidth > 1200) {

            crossAxisCount = 4; // For very wide desktops

          } else if (constraints.maxWidth > 900) {

            crossAxisCount = 3; // For standard desktops/large tablets

          } else if (constraints.maxWidth > 600) {

            crossAxisCount = 2; // For smaller tablets/larger phones

          } else {

            crossAxisCount = 1; // For phones

          }

          final horizontalPadding = constraints.maxWidth \* 0.05;

          final verticalPadding = constraints.maxWidth \*

              0.02; // Adjust vertical padding based on width

          late double titleFontSize;

          late double cardHorizontalPadding;

          late double cardVerticalPadding;

          late double emailFontSize;

          late double messageFontSize;

          late double spacingBetween;

          if (constraints.maxWidth > 1200) {

            titleFontSize = constraints.maxWidth \* 0.04;

            cardHorizontalPadding = constraints.maxWidth \* 0.01;

            cardVerticalPadding = constraints.maxWidth \* 0.010;

            emailFontSize = constraints.maxWidth \* 0.017;

            messageFontSize = constraints.maxWidth \* 0.01;

            spacingBetween = constraints.maxWidth \* 0.01;

          } else if (constraints.maxWidth > 900) {

            titleFontSize = constraints.maxWidth \* 0.02;

            cardHorizontalPadding = constraints.maxWidth \* 0.01;

            cardVerticalPadding = constraints.maxWidth \* 0.010;

            emailFontSize = constraints.maxWidth \* 0.02;

            messageFontSize = constraints.maxWidth \* 0.02;

            spacingBetween = constraints.maxWidth \* 0.01;

          } else if (constraints.maxWidth > 600) {

            titleFontSize = constraints.maxWidth \* 0.04;

            cardHorizontalPadding = constraints.maxWidth \* 0.02;

            cardVerticalPadding = constraints.maxWidth \* 0.015;

            emailFontSize = constraints.maxWidth \* 0.04;

            messageFontSize = constraints.maxWidth \* 0.03;

            spacingBetween = constraints.maxWidth \* 0.01;

          } else {

            titleFontSize = constraints.maxWidth \* 0.04;

            cardHorizontalPadding = constraints.maxWidth \* 0.02;

            cardVerticalPadding = constraints.maxWidth \* 0.015;

            emailFontSize = constraints.maxWidth \* 0.05;

            messageFontSize = constraints.maxWidth \* 0.04;

            spacingBetween = constraints.maxWidth \* 0.01;

          }

          return Padding(

            padding: EdgeInsets.symmetric(

                horizontal: horizontalPadding, vertical: verticalPadding),

            child: StreamBuilder<QuerySnapshot>(

              stream: FirebaseFirestore.instance

                  .collection('contacts')

                  .orderBy('timestamp', descending: true)

                  .snapshots(),

              builder: (context, snapshot) {

                if (!snapshot.hasData) {

                  return const Center(child: CircularProgressIndicator());

                }

                final allDocs = snapshot.data!.docs;

                if (allDocs.isEmpty) {

                  return const Center(

                      child: Text('No testimonials available.',

                          style: TextStyle(color: Colors.white)));

                }

                final totalPages = (allDocs.length / \_itemsPerPage).ceil();

                final currentDocs = allDocs

                    .skip(\_currentPage \* \_itemsPerPage)

                    .take(\_itemsPerPage)

                    .toList();

                return Column(

                  children: [

                    Text(

                      'Testimonials',

                      style: TextStyle(

                        color: Colors.white,

                        fontSize: titleFontSize,

                        fontWeight: FontWeight.bold,

                      ),

                      textAlign: TextAlign.center,

                    ),

                    SizedBox(height: spacingBetween),

                    const Divider(color: Colors.white70, thickness: 1),

                    SizedBox(height: spacingBetween \* 2),

                    Expanded(

                      child: GridView.count(

                        crossAxisCount: crossAxisCount,

                        crossAxisSpacing: spacingBetween \* 4,

                        mainAxisSpacing: spacingBetween \* 4,

                        childAspectRatio: constraints.maxWidth > 1200

                            ? 1.10

                            : constraints.maxWidth > 600

                                ? 0.9

                                : 1.8,

                        children: currentDocs.map((doc) {

                          final email = doc['email'] ?? 'No email';

                          final message = doc['message'] ?? 'No message';

                          return Container(

                            decoration: BoxDecoration(

                              color: const Color.fromARGB(255, 30, 15, 40),

                              borderRadius: BorderRadius.circular(16),

                              boxShadow: [

                                BoxShadow(

                                  color: Colors.white.withOpacity(0.1),

                                  blurRadius: 8,

                                  offset: const Offset(0, 4),

                                ),

                              ],

                            ),

                            padding: EdgeInsets.symmetric(

                                horizontal: cardHorizontalPadding,

                                vertical: cardVerticalPadding),

                            child: Column(

                              crossAxisAlignment: CrossAxisAlignment.start,

                              children: [

                                Text(

                                  email,

                                  style: TextStyle(

                                    color: Colors.lightBlueAccent,

                                    fontWeight: FontWeight.bold,

                                    fontSize: emailFontSize,

                                    letterSpacing: 1.0,

                                  ),

                                ),

                                SizedBox(height: spacingBetween \* 3),

                                Text(

                                  message,

                                  style: TextStyle(

                                    color: Colors.white,

                                    fontSize: messageFontSize,

                                    height: 1.5,

                                  ),

                                  maxLines: 5,

                                  overflow: TextOverflow.ellipsis,

                                ),

                              ],

                            ),

                          );

                        }).toList(),

                      ),

                    ),

                    SizedBox(height: spacingBetween \* 2),

                    Row(

                      mainAxisAlignment: MainAxisAlignment.center,

                      children: [

                        IconButton(

                          icon:

                              const Icon(Icons.arrow\_back, color: Colors.white),

                          onPressed: \_currentPage > 0

                              ? () => setState(() => \_currentPage--)

                              : null,

                        ),

                        const SizedBox(width: 10),

                        Text('Page ${\_currentPage + 1} of $totalPages',

                            style: const TextStyle(color: Colors.white)),

                        const SizedBox(width: 10),

                        IconButton(

                          icon: const Icon(Icons.arrow\_forward,

                              color: Colors.white),

                          onPressed: (\_currentPage + 1) < totalPages

                              ? () => setState(() => \_currentPage++)

                              : null,

                        ),

                      ],

                    ),

                    SizedBox(height: spacingBetween \* 1),

                     const Divider(

                        thickness: 1,

                        color: Color.fromARGB(59, 243, 239, 239)),

                    const CustomFooter(),

                  ],

                );

              },

            ),

          );

        },

      ),

    );

  }

  void \_showProfileDialog(BuildContext context, User user) {

    showDialog(

      context: context,

      builder: (\_) {

        return AlertDialog(

          title: const Text('Profile Details'),

          content: Column(

            mainAxisSize: MainAxisSize.min,

            children: [

              if (user.email != null) Text('Email: ${user.email}'),

              if (!user.emailVerified)

                ElevatedButton(

                  onPressed: () {

                    context

                        .read<FirebaseAuthMethods>()

                        .sendEmailVerification(context);

                  },

                  child: const Text('Verify Email'),

                ),

            ],

          ),

          actions: [

            TextButton(

              onPressed: () => Navigator.pop(context),

              child: const Text('Close'),

            ),

          ],

        );

      },

    );

  }

  Widget \_buildDrawer() {

    final user = FirebaseAuth.instance.currentUser;

    return Drawer(

      backgroundColor: const Color.fromARGB(255, 18, 6, 29),

      child: ListView(

        padding: EdgeInsets.zero,

        children: [

          const DrawerHeader(

            decoration: BoxDecoration(

              gradient: LinearGradient(

                colors: [

                  Color.fromARGB(255, 44, 1, 49),

                  Color.fromARGB(99, 89, 2, 97),

                ],

              ),

            ),

            child: Text('Menu',

                style: TextStyle(color: Colors.white, fontSize: 24)),

          ),

          ListTile(

            leading: const Icon(Icons.home, color: Colors.white),

            title: const Text('Home', style: TextStyle(color: Colors.white)),

            onTap: () {

              Navigator.pop(context);

              Navigator.pushReplacement(

                context,

                MaterialPageRoute(builder: (\_) => const HomeScreen()),

              );

            },

          ),

          ExpansionTile(

            leading: const Icon(Icons.settings, color: Colors.white),

            title:

                const Text('Settings', style: TextStyle(color: Colors.white)),

            iconColor: Colors.white,

            collapsedIconColor: Colors.white,

            children: [

              \_buildDrawerItem(

                icon: Icons.notifications\_active,

                text: 'Rate Alerts',

                screen: const RateAlertsScreen(),

              ),

              \_buildDrawerItem(

                icon: Icons.trending\_up,

                text: 'Currency News',

                screen: const CurrencyNewsScreen(),

              ),

              \_buildDrawerItem(

                icon: Icons.support\_agent,

                text: 'Help Center',

                screen: const HelpCenterScreen(),

              ),

              if (user != null)

                \_buildDrawerItem(

                  icon: Icons.history,

                  text: 'Conversion History',

                  screen: const ConversionHistoryScreen(),

                ),

              if (user != null)

                \_buildDrawerItem(

                  icon: Icons.currency\_exchange,

                  text: 'Default Currency',

                  screen: const DefaultCurrencyScreen(),

                ),

            ],

          ),

          ListTile(

            leading: const Icon(Icons.info, color: Colors.white),

            title: const Text('Blog', style: TextStyle(color: Colors.white)),

            onTap: () {

              Navigator.pop(context);

              Navigator.pushReplacement(

                context,

            MaterialPageRoute(

            builder: (\_) => BlogScreen(post: BlogScreen.blogPosts[0]),

  ),

     );

            },

          ),

          ListTile(

            leading: const Icon(Icons.star, color: Colors.white),

            title: const Text('Testimonial',

                style: TextStyle(color: Colors.white)),

            onTap: () {

              Navigator.pop(context);

              Navigator.pushReplacement(

                context,

                MaterialPageRoute(builder: (\_) => const TestimonialPage()),

              );

            },

          ),

          ListTile(

            leading: const Icon(Icons.call, color: Colors.white),

            title: const Text('Contact', style: TextStyle(color: Colors.white)),

            onTap: () {

              Navigator.pop(context);

              Navigator.pushReplacement(

                context,

                MaterialPageRoute(builder: (\_) => const ContactPage()),

              );

            },

          ),

        ],

      ),

    );

  }

  Widget \_buildDrawerItem({

    required IconData icon,

    required String text,

    required Widget screen,

  }) {

    return ListTile(

      leading: Icon(icon, color: Colors.white),

      title: Text(text, style: const TextStyle(color: Colors.white)),

      onTap: () {

        Navigator.pop(context);

        Navigator.pushReplacement(

          context,

          MaterialPageRoute(builder: (\_) => screen),

        );

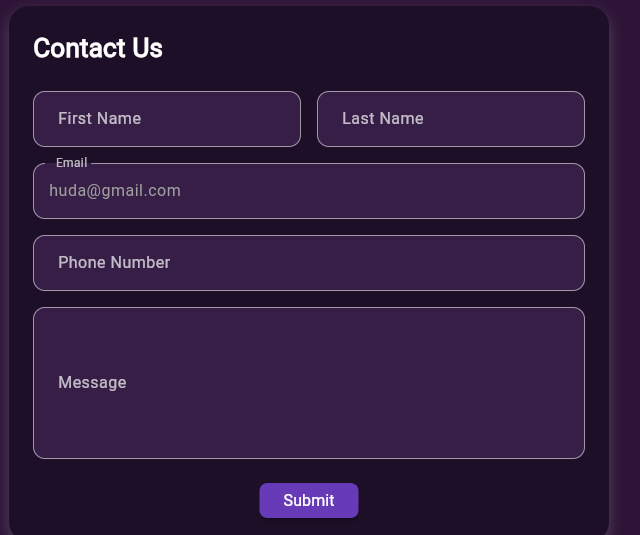
      },

    );

  }

}

**Contact Us**

****

class ContactPage extends StatefulWidget {

  const ContactPage({Key? key}) : super(key: key);

  @override

  State<ContactPage> createState() => \_ContactPageState();

}

class \_ContactPageState extends State<ContactPage> {

  final \_formKey = GlobalKey<FormState>();

  final \_firstNameController = TextEditingController();

  final \_lastNameController = TextEditingController();

  final \_phoneController = TextEditingController();

  final \_messageController = TextEditingController();

  @override

  void dispose() {

    \_firstNameController.dispose();

    \_lastNameController.dispose();

    \_phoneController.dispose();

    \_messageController.dispose();

    super.dispose();

  }

  Future<void> \_submitForm(User user) async {

    if (\_formKey.currentState!.validate()) {

      try {

        await FirebaseFirestore.instance.collection('contacts').add({

          'firstName': \_firstNameController.text,

          'lastName': \_lastNameController.text,

          'email': user.email,

          'phone': \_phoneController.text,

          'message': \_messageController.text,

          'timestamp': FieldValue.serverTimestamp(),

        });

        \_firstNameController.clear();

        \_lastNameController.clear();

        \_phoneController.clear();

        \_messageController.clear();

        ScaffoldMessenger.of(context).showSnackBar(

          const SnackBar(content: Text('Message Sent!')),

        );

        Navigator.pushReplacement(

          context,

          MaterialPageRoute(builder: (\_) => home.HomeScreen()),

        );

      } catch (e) {

        ScaffoldMessenger.of(context).showSnackBar(

          SnackBar(content: Text('Error sending message: $e')),

        );

      }

    }

  }

  @override

  Widget build(BuildContext context) {

    final user = FirebaseAuth.instance.currentUser;

    if (user == null) {

      Future.delayed(Duration.zero, () {

        Navigator.pushReplacement(

          context,

          MaterialPageRoute(builder: (\_) => const EmailPasswordLogin()),

        );

      });

      return const Scaffold(

        body: Center(child: CircularProgressIndicator()),

      );

    }

    return Scaffold(

      backgroundColor: const Color.fromARGB(255, 45, 20, 55),

      appBar: AppBar(

        backgroundColor: const Color.fromARGB(255, 18, 6, 29),

        iconTheme: const IconThemeData(color: Colors.white),

        title: Row(

          children: [

            Image.asset('assets/logo.png', height: 40),

            const SizedBox(width: 10),

            const Text('Currency App', style: TextStyle(color: Colors.white)),

          ],

        ),

        actions: [

          IconButton(

            icon: const Icon(Icons.account\_circle, color: Colors.white),

            onPressed: () => \_showProfileDialog(context, user),

          ),

          PopupMenuButton<String>(

            icon: const Icon(Icons.more\_vert, color: Colors.white),

            onSelected: (value) async {

              if (value == 'signout') {

                await context.read<FirebaseAuthMethods>().signOut(context);

                Navigator.pushReplacement(

                  context,

                  MaterialPageRoute(builder: (\_) => const EmailPasswordLogin()),

                );

              } else if (value == 'delete') {

                await context

                    .read<FirebaseAuthMethods>()

                    .deleteAccount(context);

              }

            },

            itemBuilder: (context) => const [

              PopupMenuItem(value: 'signout', child: Text('Sign Out')),

              PopupMenuItem(value: 'delete', child: Text('Delete Account')),

            ],

          ),

        ],

      ),

      drawer: \_buildDrawer(user),

      body: SingleChildScrollView(

        padding: const EdgeInsets.all(20),

        child: Column(

          children: [

            Center(

              child: Container(

                padding: const EdgeInsets.all(24),

                width: 600,

                decoration: BoxDecoration(

                  color: const Color.fromARGB(255, 30, 15, 40),

                  borderRadius: BorderRadius.circular(20),

                  boxShadow: [

                    BoxShadow(

                      color: Colors.white.withOpacity(0.2),

                      blurRadius: 12,

                      offset: const Offset(0, 6),

                    ),

                  ],

                ),

                child: Form(

                  key: \_formKey,

                  child: Column(

                    crossAxisAlignment: CrossAxisAlignment.start,

                    children: [

                      const Text(

                        'Contact Us',

                        style: TextStyle(

                          color: Colors.white,

                          fontSize: 26,

                          fontWeight: FontWeight.bold,

                        ),

                      ),

                      const SizedBox(height: 24),

                      Row(

                        children: [

                          Expanded(

                            child: TextFormField(

                              controller: \_firstNameController,

                              style: const TextStyle(color: Colors.white),

                              decoration: \_inputDecoration('First Name'),

                              validator: (value) =>

                                  value == null || value.isEmpty

                                      ? 'Enter your first name'

                                      : null,

                            ),

                          ),

                          const SizedBox(width: 16),

                          Expanded(

                            child: TextFormField(

                              controller: \_lastNameController,

                              style: const TextStyle(color: Colors.white),

                              decoration: \_inputDecoration('Last Name'),

                              validator: (value) =>

                                  value == null || value.isEmpty

                                      ? 'Enter your last name'

                                      : null,

                            ),

                          ),

                        ],

                      ),

                      const SizedBox(height: 16),

                      TextFormField(

                        initialValue: user.email,

                        readOnly: true,

                        style: const TextStyle(color: Colors.grey),

                        decoration: \_inputDecoration('Email'),

                      ),

                      const SizedBox(height: 16),

                      TextFormField(

                        controller: \_phoneController,

                        keyboardType: TextInputType.phone,

                        style: const TextStyle(color: Colors.white),

                        decoration: \_inputDecoration('Phone Number'),

                        validator: (value) => value == null || value.isEmpty

                            ? 'Enter your phone number'

                            : null,

                      ),

                      const SizedBox(height: 16),

                      TextFormField(

                        controller: \_messageController,

                        style: const TextStyle(color: Colors.white),

                        decoration: \_inputDecoration("Message"),

                        maxLines: 5,

                        validator: (value) => value == null || value.isEmpty

                            ? 'Enter your message'

                            : null,

                      ),

                      const SizedBox(height: 24),

                      Center(

                        child: CustomButton(

                          label: 'Submit',

                          onPressed: () => \_submitForm(user),

                        ),

                      ),

                    ],

                  ),

                ),

              ),

            ),

            const SizedBox(height: 32),

            const Divider(

                thickness: 1, color: Color.fromARGB(57, 250, 246, 246)),

            const CustomFooter(),

          ],

        ),

      ),

    );

  }

  InputDecoration \_inputDecoration(String label) {

    return InputDecoration(

      labelText: '  $label',

      labelStyle: const TextStyle(color: Colors.white70),

      filled: true,

      fillColor: const Color.fromARGB(255, 55, 30, 70),

      contentPadding: const EdgeInsets.symmetric(vertical: 20, horizontal: 16),

      enabledBorder: OutlineInputBorder(

        borderRadius: BorderRadius.circular(12),

        borderSide: const BorderSide(color: Colors.white54),

      ),

      focusedBorder: OutlineInputBorder(

        borderRadius: BorderRadius.circular(12),

        borderSide: const BorderSide(color: Colors.white),

      ),

      border: OutlineInputBorder(

        borderRadius: BorderRadius.circular(12),

      ),

    );

  }

  void \_showProfileDialog(BuildContext context, User user) {

    showDialog(

      context: context,

      builder: (\_) => AlertDialog(

        title: const Text('Profile Details'),

        content: Column(

          mainAxisSize: MainAxisSize.min,

          children: [

            if (user.email != null) Text('Email: ${user.email}'),

            if (!user.emailVerified)

              CustomButton(

                label: 'Verify Email',

                onPressed: () {

                  context

                      .read<FirebaseAuthMethods>()

                      .sendEmailVerification(context);

                },

              ),

          ],

        ),

        actions: [

          TextButton(

            onPressed: () => Navigator.pop(context),

            child: const Text('Close'),

          ),

        ],

      ),

    );

  }

  Widget \_buildDrawer(User? user) {

    return Drawer(

      backgroundColor: const Color.fromARGB(255, 18, 6, 29),

      child: ListView(

        padding: EdgeInsets.zero,

        children: [

          const DrawerHeader(

            decoration: BoxDecoration(

              gradient: LinearGradient(

                colors: [

                  Color.fromARGB(255, 44, 1, 49),

                  Color.fromARGB(99, 89, 2, 97)

                ],

              ),

            ),

            child: Text('Menu',

                style: TextStyle(color: Colors.white, fontSize: 24)),

          ),

          \_buildDrawerItem(

              icon: Icons.home, text: 'Home', screen: home.HomeScreen()),

          if (user != null)

            ExpansionTile(

              leading: const Icon(Icons.settings, color: Colors.white),

              title:

                  const Text('Settings', style: TextStyle(color: Colors.white)),

              iconColor: Colors.white,

              collapsedIconColor: Colors.white,

              children: [

                \_buildDrawerItem(

                  icon: Icons.notifications\_active,

                  text: 'Rate Alerts',

                  screen: const RateAlertsScreen(),

                ),

                \_buildDrawerItem(

                  icon: Icons.trending\_up,

                  text: 'Currency News',

                  screen: const CurrencyNewsScreen(),

                ),

                \_buildDrawerItem(

                  icon: Icons.support\_agent,

                  text: 'Help Center',

                  screen: const HelpCenterScreen(),

                ),

                \_buildDrawerItem(

                  icon: Icons.history,

                  text: 'Conversion History',

                  screen: const ConversionHistoryScreen(),

                ),

                \_buildDrawerItem(

                  icon: Icons.currency\_exchange,

                  text: 'Default Currency',

                  screen: const DefaultCurrencyScreen(),

                ),

              ],

            ),

          \_buildDrawerItem(

            icon: Icons.reviews,

            text: 'Testimonial',

            screen: const TestimonialPage(),

          ),

          \_buildDrawerItem(

            icon: Icons.contact\_page,

            text: 'Contact',

            screen: const ContactPage(),

          ),

        ],

      ),

    );

  }

  Widget \_buildDrawerItem(

      {required IconData icon, required String text, required Widget screen}) {

    return ListTile(

      leading: Icon(icon, color: Colors.white),

      title: Text(text, style: const TextStyle(color: Colors.white)),

      onTap: () {

        Navigator.pop(context);

        Navigator.pushReplacement(

          context,

          MaterialPageRoute(builder: (\_) => screen),

        );

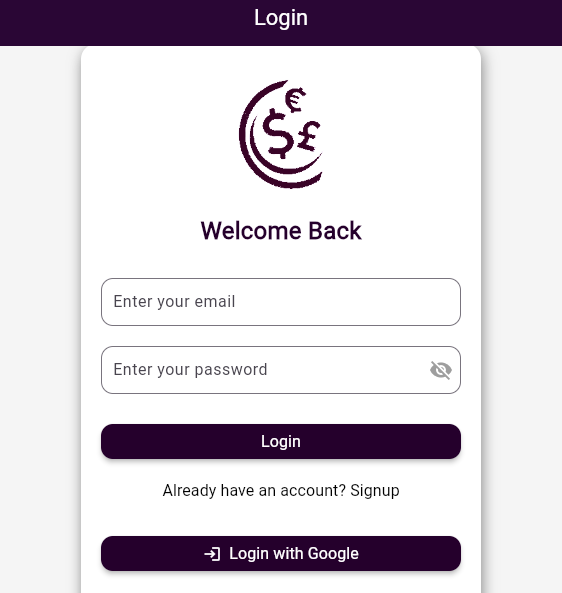
      },

    );

  }

}

**Login**

****

class LoginScreen extends StatefulWidget {

  // static String routeName;

  const LoginScreen({Key? key}) : super(key: key);

  @override

  State<LoginScreen> createState() => \_LoginScreenState();

}

class CustomButton extends StatelessWidget {

  final String label;

  final VoidCallback onPressed;

  const CustomButton({

    Key? key,

    required this.label,

    required this.onPressed,

  }) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return ElevatedButton(

      onPressed: onPressed, // Correct usage of onPressed

      style: ElevatedButton.styleFrom(

        backgroundColor: Colors.blue, // Set your button color here

        padding: const EdgeInsets.symmetric(horizontal: 24, vertical: 12),

      ),

      child: Text(label), // Display label on button

    );

  }

}

class \_LoginScreenState extends State<LoginScreen> {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: SafeArea(

        child: Column(

          children: [

            CustomButton(

              label: 'Email/Password Sign Up', // Correct parameter

              onPressed: () {

                Navigator.pushNamed(context, EmailPasswordSignup.routeName);

              },

            ),

            CustomButton(

              label: 'Email/Password Login', // Correct parameter

              onPressed: () {

                Navigator.pushNamed(context, EmailPasswordLogin.routeName);

              },

            ),

            CustomButton(

              label: 'Phone Sign In', // Correct parameter

              onPressed: () {

                Navigator.pushNamed(context, PhoneScreen.routeName);

              },

            ),

            CustomButton(

              label: 'Google Sign In', // Correct parameter

              onPressed: () {

                context.read<FirebaseAuthMethods>().signInWithGoogle(context);

              },

            ),

            CustomButton(

              label: 'Facebook Sign In', // Correct parameter

              onPressed: () {

                context.read<FirebaseAuthMethods>().signInWithFacebook(context);

              },

            ),

            CustomButton(

              label: 'Anonymous Sign In', // Correct parameter

              onPressed: () {

                context.read<FirebaseAuthMethods>().signInAnonymously(context);

              },

            ),

          ],

        ),

      ),

    );

  }

}

**Login with Email**

class CustomTextField extends StatelessWidget {

  final TextEditingController controller;

  final String hintText;

  final bool obscureText;

  final Widget? suffixIcon;

  const CustomTextField({

    Key? key,

    required this.controller,

    required this.hintText,

    this.obscureText = false, // Default value is false for email fields

    this.suffixIcon,

  }) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return TextField(

      controller: controller,

      obscureText: obscureText, // To hide/show text

      decoration: InputDecoration(

        hintText: hintText,

        suffixIcon: suffixIcon, // Optional icon for password field

        border: OutlineInputBorder(

          borderRadius: BorderRadius.circular(12),

        ),

      ),

    );

  }

}

class EmailPasswordLogin extends StatefulWidget {

  static String routeName = '/login-email-password';

  const EmailPasswordLogin({Key? key}) : super(key: key);

  @override

  \_EmailPasswordLoginState createState() => \_EmailPasswordLoginState();

}

class \_EmailPasswordLoginState extends State<EmailPasswordLogin> {

  final TextEditingController emailController = TextEditingController();

  final TextEditingController passwordController = TextEditingController();

  bool \_obscureText = true; // For toggling password visibility

  void loginUser() async {

    try {

      await context.read<FirebaseAuthMethods>().loginWithEmail(

            email: emailController.text,

            password: passwordController.text,

            context: context,

          );

      Navigator.pushReplacement(

        context,

        MaterialPageRoute(builder: (context) => const HomeScreen()),

      );

    } catch (e) {

      print('Error during login: $e');

      ScaffoldMessenger.of(context).showSnackBar(

        const SnackBar(content: Text('Failed to login. Please try again.')),

      );

    }

  }

  void goToSignupScreen() {

    // Navigate to Signup Screen

    Navigator.pushNamed(context, '/signup-email-password');

  }

  @override

  Widget build(BuildContext context) {

    final themeColor = const Color.fromARGB(255, 37, 0, 44);

    return Scaffold(

      appBar: AppBar(

        title: const Text(

          "Login",

          style: TextStyle(color: Colors.white),

        ),

        backgroundColor: themeColor,

        centerTitle: true,

        iconTheme: const IconThemeData(color: Colors.white),

      ),

      backgroundColor: Colors.grey[100],

      body: LayoutBuilder(

        builder: (context, constraints) {

          return SingleChildScrollView(

            child: ConstrainedBox(

              constraints: BoxConstraints(minHeight: constraints.maxHeight),

              child: Center(

                child: Padding(

                  padding: const EdgeInsets.all(16),

                  child: Card(

                    shape: RoundedRectangleBorder(

                      borderRadius: BorderRadius.circular(20),

                    ),

                    elevation: 10,

                    shadowColor: Colors.grey.withOpacity(0.5),

                    child: Container(

                      width: double.infinity,

                      constraints: const BoxConstraints(maxWidth: 400),

                      padding: const EdgeInsets.symmetric(

                          horizontal: 20, vertical: 30),

                      decoration: BoxDecoration(

                        color: Colors.white,

                        borderRadius: BorderRadius.circular(15),

                        boxShadow: const [

                          BoxShadow(

                            color: Colors.black26,

                            blurRadius: 10,

                            spreadRadius: 2,

                            offset: Offset(2, 4),

                          ),

                        ],

                      ),

                      child: Column(

                        mainAxisSize: MainAxisSize.min,

                        children: [

                          // Logo

                          Image.asset(

                            'assets/logoo.png',

                            height: 120,

                            width: 160,

                            fit: BoxFit.contain,

                          ),

                          const SizedBox(height: 20),

                          // Title

                          Text(

                            "Welcome Back",

                            style: TextStyle(

                              fontSize: 24,

                              fontWeight: FontWeight.bold,

                              color: themeColor,

                            ),

                          ),

                          const SizedBox(height: 30),

                          // Email

                          CustomTextField(

                            controller: emailController,

                            hintText: 'Enter your email',

                            obscureText:

                                false, // Email field does not need to be obscure

                          ),

                          const SizedBox(height: 20),

                          // Password with eye icon

                          CustomTextField(

                            controller: passwordController,

                            hintText: 'Enter your password',

                            obscureText: \_obscureText, // Toggle visibility

                            suffixIcon: IconButton(

                              icon: Icon(

                                \_obscureText

                                    ? Icons.visibility\_off

                                    : Icons.visibility,

                                color: Colors.grey,

                              ),

                              onPressed: () {

                                setState(() {

                                  \_obscureText =

                                      !\_obscureText; // Toggle visibility

                                });

                              },

                            ),

                          ),

                          const SizedBox(height: 30),

                          // Login Button

                          SizedBox(

                            width: double.infinity,

                            child: ElevatedButton(

                              onPressed: loginUser,

                              style: ElevatedButton.styleFrom(

                                backgroundColor: themeColor,

                                padding: const EdgeInsets.symmetric(

                                  vertical: 14,

                                ),

                                shape: RoundedRectangleBorder(

                                  borderRadius: BorderRadius.circular(12),

                                ),

                                elevation: 4,

                              ),

                              child: const Text(

                                "Login",

                                style: TextStyle(

                                  color: Colors.white,

                                  fontSize: 16,

                                ),

                              ),

                            ),

                          ),

                          const SizedBox(height: 15),

                          // "Don't have an account?" link

                          TextButton(

                            onPressed: () {

                              Navigator.pushReplacementNamed(

                                  context, '/signup-email-password');

                            },

                            child: const Text(

                              "Already have an account? Signup",

                              style: TextStyle(

                                fontSize: 16,

                                color: Colors.black,

                              ),

                            ),

                          ),

                          const SizedBox(height: 30),

                          // Login with Google button

                          SizedBox(

                            width: double.infinity,

                            child: ElevatedButton.icon(

                              onPressed: () {

                                context

                                    .read<FirebaseAuthMethods>()

                                    .signInWithGoogle(context);

                              },

                              style: ElevatedButton.styleFrom(

                                backgroundColor:

                                    const Color.fromARGB(255, 37, 0, 44),

                                padding:

                                    const EdgeInsets.symmetric(vertical: 14),

                                shape: RoundedRectangleBorder(

                                  borderRadius: BorderRadius.circular(12),

                                ),

                                elevation: 4,

                              ),

                              icon:

                                  const Icon(Icons.login, color: Colors.white),

                              label: const Text(

                                "Login with Google",

                                style: TextStyle(

                                  color: Colors.white,

                                  fontSize: 16,

                                ),

                              ),

                            ),

                          ),

                          const SizedBox(height: 20),

                          // Login with Phone Number button

                          // SizedBox(

                          //   width: double.infinity,

                          //   child: ElevatedButton.icon(

                          //     onPressed: () {

                          //       Navigator.pushNamed(

                          //           context, PhoneScreen.routeName);

                          //     },

                          //     style: ElevatedButton.styleFrom(

                          //       backgroundColor:

                          //           const Color.fromARGB(255, 37, 0, 44),

                          //       padding:

                          //           const EdgeInsets.symmetric(vertical: 14),

                          //       shape: RoundedRectangleBorder(

                          //         borderRadius: BorderRadius.circular(12),

                          //       ),

                          //       elevation: 4,

                          //     ),

                          //     icon:

                          //         const Icon(Icons.phone, color: Colors.white),

                          //     label: const Text(

                          //       "Login with Phone Number",

                          //       style: TextStyle(

                          //         color: Colors.white,

                          //         fontSize: 16,

                          //       ),

                          //     ),

                          //   ),

                          // ),

                        ],

                      ),

                    ),

                  ),

                ),

              ),

            ),

          );

        },

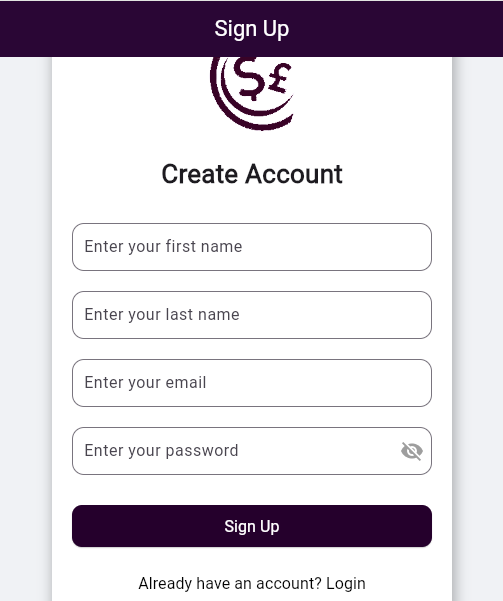
      ),

    );

  }

}

**Signup**

****

class EmailPasswordSignup extends StatefulWidget {

  static String routeName = '/signup-email-password';

  const EmailPasswordSignup({Key? key}) : super(key: key);

  @override

  \_EmailPasswordSignupState createState() => \_EmailPasswordSignupState();

}

class \_EmailPasswordSignupState extends State<EmailPasswordSignup> {

  final TextEditingController emailController = TextEditingController();

  final TextEditingController passwordController = TextEditingController();

  final TextEditingController firstNameController = TextEditingController();

  final TextEditingController lastNameController = TextEditingController();

  bool \_obscureText = true; // Password visibility toggle

  void signUpUser() async {

    await context.read<FirebaseAuthMethods>().signUpWithEmail(

          email: emailController.text.trim(),

          password: passwordController.text.trim(),

          context: context,

          firstName: firstNameController.text.trim(),

          lastName: lastNameController.text.trim(),

        );

    if (context.read<FirebaseAuthMethods>().user != null) {

      Navigator.pushReplacement(

        context,

        MaterialPageRoute(builder: (context) => const HomeScreen()),

      );

    }

  }

  @override

  Widget build(BuildContext context) {

    final themeColor = const Color.fromARGB(255, 37, 0, 44);

    return Scaffold(

      appBar: AppBar(

        title: const Text("Sign Up", style: TextStyle(color: Colors.white)),

        backgroundColor: themeColor,

        centerTitle: true,

        iconTheme: const IconThemeData(color: Colors.white),

      ),

      body: Center(

        child: SingleChildScrollView(

          padding: const EdgeInsets.all(20),

          child: Container(

            width: double.infinity,

            constraints: const BoxConstraints(maxWidth: 400),

            padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 30),

            decoration: BoxDecoration(

              color: Colors.white,

              borderRadius: BorderRadius.circular(15),

              boxShadow: const [

                BoxShadow(

                  color: Colors.black26,

                  blurRadius: 10,

                  spreadRadius: 2,

                  offset: Offset(2, 4),

                ),

              ],

            ),

            child: Column(

              mainAxisSize: MainAxisSize.min,

              children: [

                Transform.translate(

                  offset: const Offset(0, -10),

                  child: Image.asset(

                    'assets/logoo.png',

                    height: 120,

                    width: 160,

                    fit: BoxFit.contain,

                  ),

                ),

                const SizedBox(height: 10),

                const Text(

                  "Create Account",

                  style: TextStyle(

                    fontSize: 26,

                    fontWeight: FontWeight.bold,

                  ),

                ),

                const SizedBox(height: 30),

                // First Name Field

                CustomTextField(

                  controller: firstNameController,

                  hintText: 'Enter your first name',

                ),

                const SizedBox(height: 20),

                // Last Name Field

                CustomTextField(

                  controller: lastNameController,

                  hintText: 'Enter your last name',

                ),

                const SizedBox(height: 20),

                // Email Field

                CustomTextField(

                  controller: emailController,

                  hintText: 'Enter your email',

                  obscureText: false,

                ),

                const SizedBox(height: 20),

                // Password Field

                CustomTextField(

                  controller: passwordController,

                  hintText: 'Enter your password',

                  obscureText: \_obscureText,

                  suffixIcon: IconButton(

                    icon: Icon(

                      \_obscureText ? Icons.visibility\_off : Icons.visibility,

                      color: Colors.grey,

                    ),

                    onPressed: () {

                      setState(() {

                        \_obscureText = !\_obscureText;

                      });

                    },

                  ),

                ),

                const SizedBox(height: 30),

                // Sign Up Button

                ElevatedButton(

                  onPressed: signUpUser,

                  style: ElevatedButton.styleFrom(

                    backgroundColor: themeColor,

                    minimumSize: const Size(double.infinity, 50),

                    shape: RoundedRectangleBorder(

                      borderRadius: BorderRadius.circular(10),

                    ),

                  ),

                  child: const Text(

                    "Sign Up",

                    style: TextStyle(fontSize: 16, color: Colors.white),

                  ),

                ),

                const SizedBox(height: 20),

                // Login Button

                TextButton(

                  onPressed: () {

                    Navigator.pushReplacementNamed(

                        context, '/login-email-password');

                  },

                  child: const Text(

                    "Already have an account? Login",

                    style: TextStyle(

                      fontSize: 16,

                      color: Colors.black,

                    ),

                  ),

                ),

                //     const SizedBox(height: 20),

                //     // Login with Google Button

                //     CustomButton(

                //       label: 'Login with Google',

                //       onPressed: () {

                //         context.read<FirebaseAuthMethods>().signInWithGoogle(context);

                //       },

                //     ),

                //     const SizedBox(height: 10),

                //     // Login with Phone Button

                //      CustomButton(

                //   label: 'Phone Sign In', // Correct parameter

                //   onPressed: () {

                //     Navigator.pushNamed(context, PhoneScreen.routeName);

                //   },

                // ),

                const SizedBox(height: 30),

                // Login with Google button

                SizedBox(

                  width: double.infinity,

                  child: ElevatedButton.icon(

                    onPressed: () {

                      context

                          .read<FirebaseAuthMethods>()

                          .signInWithGoogle(context);

                    },

                    style: ElevatedButton.styleFrom(

                      backgroundColor: Colors.red,

                      padding: const EdgeInsets.symmetric(vertical: 14),

                      shape: RoundedRectangleBorder(

                        borderRadius: BorderRadius.circular(12),

                      ),

                      elevation: 4,

                    ),

                    icon: const Icon(Icons.login, color: Colors.white),

                    label: const Text(

                      "Login with Google",

                      style: TextStyle(

                        color: Colors.white,

                        fontSize: 16,

                      ),

                    ),

                  ),

                ),

                const SizedBox(height: 20),

                // Login with Phone Number button

                // SizedBox(

                //   width: double.infinity,

                //   child: ElevatedButton.icon(

                //     onPressed: () {

                //       Navigator.pushNamed(context, PhoneScreen.routeName);

                //     },

                //     style: ElevatedButton.styleFrom(

                //       backgroundColor: Colors.blueAccent,

                //       padding: const EdgeInsets.symmetric(vertical: 14),

                //       shape: RoundedRectangleBorder(

                //         borderRadius: BorderRadius.circular(12),

                //       ),

                //       elevation: 4,

                //     ),

                //     icon: const Icon(Icons.phone, color: Colors.white),

                //     label: const Text(

                //       "Login with Phone Number",

                //       style: TextStyle(

                //         color: Colors.white,

                //         fontSize: 16,

                //       ),

                //     ),

                //   ),

                // ),

                // CustomButton(

                //   label: 'Login with Phone Number',

                //   onPressed: () {

                //     Navigator.pushNamed(context, '/phone-signup');

                //   },

                // ),

              ],

            ),

          ),

        ),

      ),

      backgroundColor: const Color(0xFFF0F2F5),

    );

  }

}

class CustomButton extends StatelessWidget {

  final String label;

  final VoidCallback onPressed;

  const CustomButton({Key? key, required this.label, required this.onPressed})

      : super(key: key);

  @override

  Widget build(BuildContext context) {

    return ElevatedButton(

      onPressed: onPressed,

      style: ElevatedButton.styleFrom(

        backgroundColor: const Color.fromARGB(255, 37, 0, 44),

        minimumSize: const Size(double.infinity, 50),

        shape: RoundedRectangleBorder(

          borderRadius: BorderRadius.circular(10),

        ),

      ),

      child: Text(

        label,

        style: const TextStyle(fontSize: 16, color: Colors.white),

      ),

    );

  }

}

**Footer**

****

class CustomFooter extends StatelessWidget {

  const CustomFooter({super.key});

  @override

  Widget build(BuildContext context) {

    return Padding(

      padding: const EdgeInsets.symmetric(vertical: 30, horizontal: 20),

      child: Column(

        children: [

          // Top Row: Logo + Navigation

          LayoutBuilder(

            builder: (context, constraints) {

              if (constraints.maxWidth < 600) {

                return Column(

                  crossAxisAlignment: CrossAxisAlignment.start,

                  children: [

                    Row(

                      children: const [

                        Icon(Icons.currency\_exchange,

                            color: Colors.white, size: 28),

                        SizedBox(width: 8),

                        Text(

                          "Currency",

                          style: TextStyle(

                            color: Colors.white,

                            fontSize: 18,

                            fontWeight: FontWeight.bold,

                            letterSpacing: 1.2,

                          ),

                        ),

                      ],

                    ),

                    const SizedBox(height: 10),

                    Wrap(

                      spacing: 10,

                      runSpacing: 8,

                      children: const [

                        \_FooterLink(text: "Home", screen: home.HomeScreen()),

                        \_FooterLink(text: "Contact", screen: ContactPage()),

                        \_FooterLink(

                            text: "Testimonials", screen: TestimonialPage()),

                        \_FooterLink(

                            text: "Rate Alerts", screen: RateAlertsScreen()),

                      ],

                    ),

                  ],

                );

              } else {

                return Row(

                  mainAxisAlignment: MainAxisAlignment.spaceBetween,

                  children: [

                    Row(

                      children: const [

                        Icon(Icons.currency\_exchange,

                            color: Colors.white, size: 28),

                        SizedBox(width: 8),

                        Text(

                          "Currency",

                          style: TextStyle(

                            color: Colors.white,

                            fontSize: 18,

                            fontWeight: FontWeight.bold,

                            letterSpacing: 1.2,

                          ),

                        ),

                      ],

                    ),

                    Row(

                      mainAxisAlignment: MainAxisAlignment.center,

                      children: const [

                        \_FooterLink(text: "Home", screen: home.HomeScreen()),

                        \_FooterLink(text: "Contact", screen: ContactPage()),

                        \_FooterLink(

                            text: "Testimonials", screen: TestimonialPage()),

                        \_FooterLink(

                            text: "Rate Alerts", screen: RateAlertsScreen()),

                      ],

                    ),

                  ],

                );

              }

            },

          ),

          const SizedBox(height: 20),

          const Divider(color: Colors.white30),

          const SizedBox(height: 15),

          // Bottom Row: Social Icons + Copyright

          LayoutBuilder(

            builder: (context, constraints) {

              if (constraints.maxWidth < 600) {

                return Column(

                  crossAxisAlignment: CrossAxisAlignment.start,

                  children: [

                    Row(

                      children: [

                        const \_SocialIcon(

                            icon: Icons.facebook, url: "https://facebook.com"),

                        IconButton(

                          icon: const Icon(Icons.email, color: Colors.white),

                          onPressed: \_launchEmail,

                        ),

                        const \_SocialIcon(

                          icon: FontAwesomeIcons.google,

                          url: "https://www.google.com",

                        ),

                      ],

                    ),

                    const SizedBox(height: 10),

                    const Text(

                      "© 2025 Currency. All rights reserved.",

                      style: TextStyle(

                        color: Colors.white70,

                        fontSize: 12,

                      ),

                    ),

                  ],

                );

              } else {

                return Row(

                  mainAxisAlignment: MainAxisAlignment.spaceBetween,

                  children: [

                    Row(

                      children: [

                        const \_SocialIcon(

                            icon: Icons.facebook, url: "https://facebook.com"),

                        IconButton(

                          icon: const Icon(Icons.email, color: Colors.white),

                          onPressed: \_launchEmail,

                        ),

                        const \_SocialIcon(

                          icon: FontAwesomeIcons.google,

                          url: "https://www.google.com",

                        ),

                      ],

                    ),

                    const Text(

                      "© 2025 Currency. All rights reserved.",

                      style: TextStyle(

                        color: Colors.white70,

                        fontSize: 12,

                      ),

                    ),

                  ],

                );

              }

            },

          ),

        ],

      ),

    );

  }

}

class \_FooterLink extends StatelessWidget {

  final String text;

  final Widget? screen;

  const \_FooterLink({required this.text, this.screen, Key? key})

      : super(key: key);

  @override

  Widget build(BuildContext context) {

    return Padding(

      padding: const EdgeInsets.symmetric(horizontal: 10),

      child: MouseRegion(

        cursor: SystemMouseCursors.click,

        child: GestureDetector(

          onTap: () {

            if (screen != null) {

              Navigator.push(

                context,

                MaterialPageRoute(builder: (\_) => screen!),

              );

            }

          },

          child: Text(

            text,

            style: const TextStyle(

              color: Colors.white70,

              fontSize: 14,

              decoration: TextDecoration.underline,

            ),

          ),

        ),

      ),

    );

  }

}

class \_SocialIcon extends StatelessWidget {

  final IconData icon;

  final String url;

  const \_SocialIcon({required this.icon, required this.url, Key? key})

      : super(key: key);

  Future<void> \_launchUrl(BuildContext context) async {

    final uri = Uri.parse(url);

    if (await canLaunchUrl(uri)) {

      await launchUrl(uri);

    } else {

      ScaffoldMessenger.of(context).showSnackBar(

        SnackBar(content: Text("Could not launch $url")),

      );

    }

  }

  @override

  Widget build(BuildContext context) {

    return Padding(

      padding: const EdgeInsets.symmetric(horizontal: 6),

      child: InkWell(

        onTap: () => \_launchUrl(context),

        child: CircleAvatar(

          backgroundColor: Colors.transparent,

          child: Icon(

            icon,

            color: Colors.white,

            size: 16,

          ),

        ),

      ),

    );

  }

}