MPL Practical 06

Name: Sharvari Kishor More
Class: D15B
Roll no.: 37

Aim: To integrate Firebase Authentication in a Flutter app.

Theory:

Firebase Authentication in Our Code
User Registration – Users can sign up using their email and password. Upon successful registration, a verification email is sent.
Email Verification – Users must verify their email before they can log in.
User Login – Only verified users can log in to the application.
Google Sign-In – Users have the option to sign in using their Google account.
Password Reset – If users forget their password, they can request a reset link via email.
Logout – Users can securely log out from their session.

Firebase Integration Steps

- Configured a Firebase project and enabled authentication services.
- Initialized Firebase in the Fluttera project and added the required dependencies.
- Implemented authentication functions in auth_service.dart to handle user sign-up, login, password reset, and logout.

Code:

```
import 'package:firebase_auth/firebase_auth.dart';

import 'package:flutter/material.dart';

import 'package:sharvari/reuseable_widgets/reusable_widgets.dart';


class ResetPassword extends StatefulWidget {

  const ResetPassword({super.key});


  @override

  _ResetPasswordState createState() => _ResetPasswordState();
```

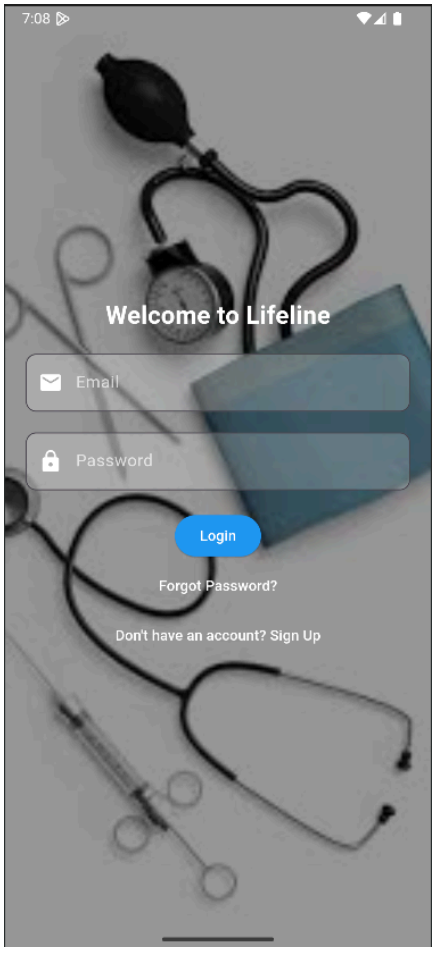
```
}

class _ResetPasswordState extends State<ResetPassword> {
  final TextEditingController _emailTextController = TextEditingController();
  final _formKey = GlobalKey<FormState>();
  bool _isLoading = false;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Stack(
        fit: StackFit.expand,
        children: [
          Image.asset("assets/images/dr1.jpg", fit: BoxFit.cover),
          Container(color: Colors.black.withOpacity(0.4)),
          Center(
            child: Padding(
              padding: const EdgeInsets.all(20),
              child: Form(
                key: _formKey,
                child: Column(
                  mainAxisSize: MainAxisSize.min,
                  children: [
                    const Text(
                      "Reset Password",
                      style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold, color: Colors.white),
```

```
            ),
            const SizedBox(height: 20),
            reuseableTextField(
              "Enter Email Id",
              Icons.email,
              false,
              _emailTextController,
                (value) {
              if (value == null || value.isEmpty) {
                return "Please enter your email";
              }
              if
(!RegExp(r"^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$").hasMatch(value)) {
                return "Please enter a valid email address";
              }
              return null;
            },
            ),
            const SizedBox(height: 20),
            ElevatedButton(
              style: ElevatedButton.styleFrom(backgroundColor: Colors.lightBlue),
              onPressed: _isLoading ? null : _resetPassword,
              child: _isLoading
                  ? const CircularProgressIndicator(color: Colors.white)
                  : const Text("Reset Password", style: TextStyle(color: Colors.white)),
            ),
            TextButton(
```

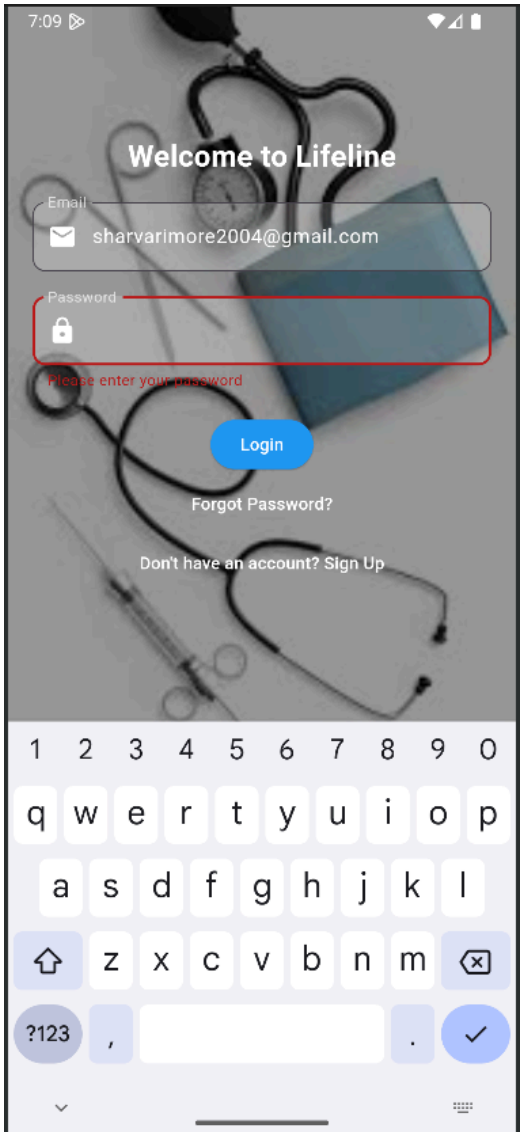
```
                onPressed: () => Navigator.pop(context),
                child: const Text("Back to Login", style: TextStyle(color: Colors.white)),
              ),
            ],
          ),
        ),
      ),
    ),
  ],
 ),
 );
}

void _resetPassword() async {
 if (!_formKey.currentState!.validate()) return;

 setState(() => _isLoading = true);
 String email = _emailTextController.text.trim();

 try {
  await FirebaseAuth.instance.sendPasswordResetEmail(email: email);
  setState(() => _isLoading = false);
  _showSnackbar("Password reset email sent successfully!");
  Navigator.of(context).pop();
 } on FirebaseAuthException catch (e) {
  setState(() => _isLoading = false);
```

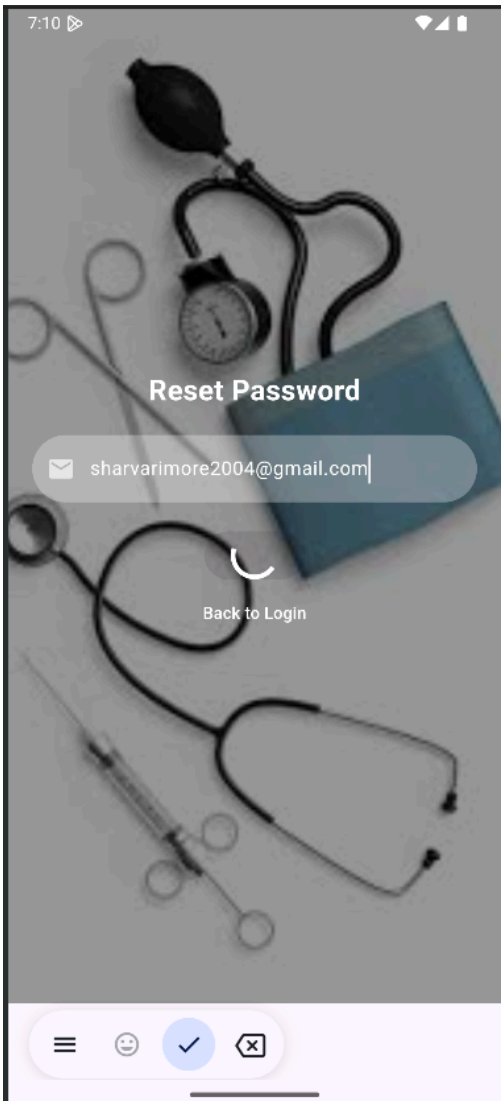
```
      _showSnackbar("Error: ${e.message}");
    } catch (e) {
      setState(() => _isLoading = false);
      _showSnackbar("An unexpected error occurred.");
    }
  }

  void _showSnackbar(String message) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(message)));
  }
}
```

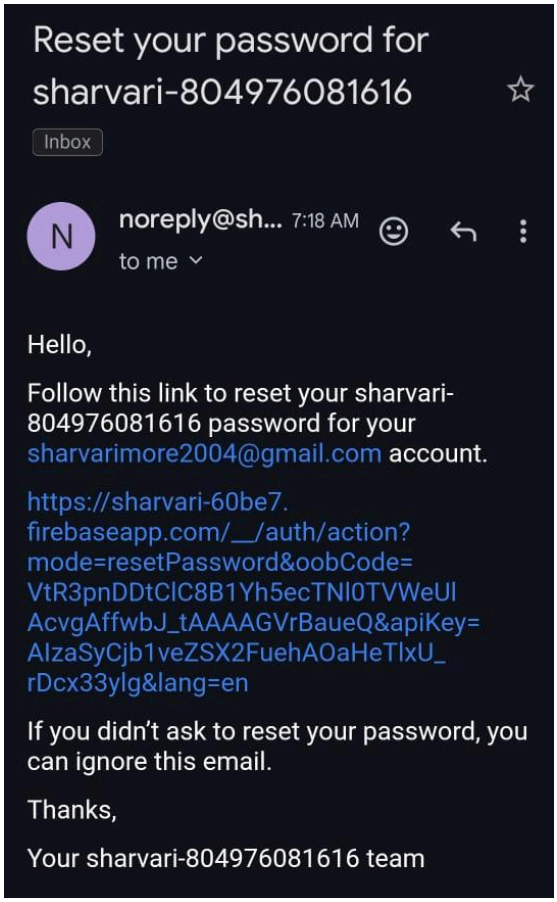
Screenshots:

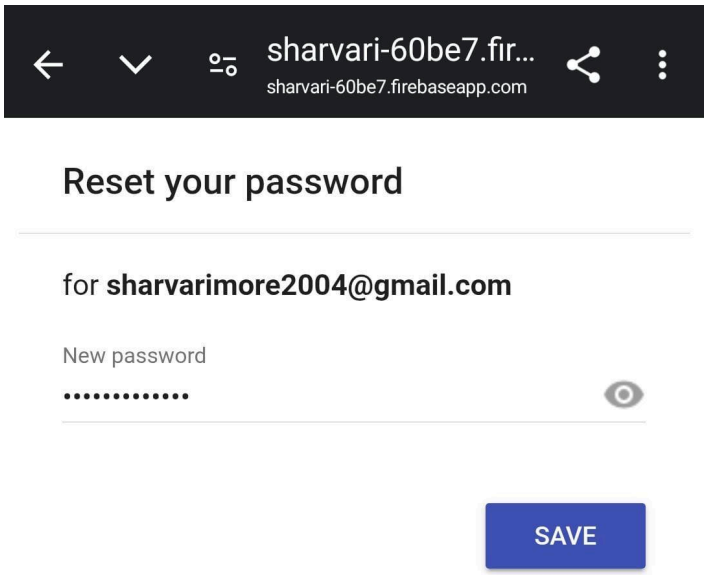


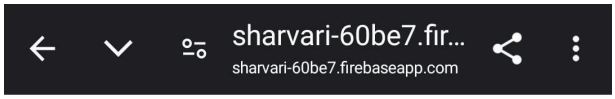
c


7:09
Welcome to Lifeline
Email
sharvarimore2004@gmail.com
Password
Please enter your password
Login
Forgot Password?
Don't have an account? Sign Up


7:10
Reset Password
sharvarimore2004@gmail.com
Back to Login

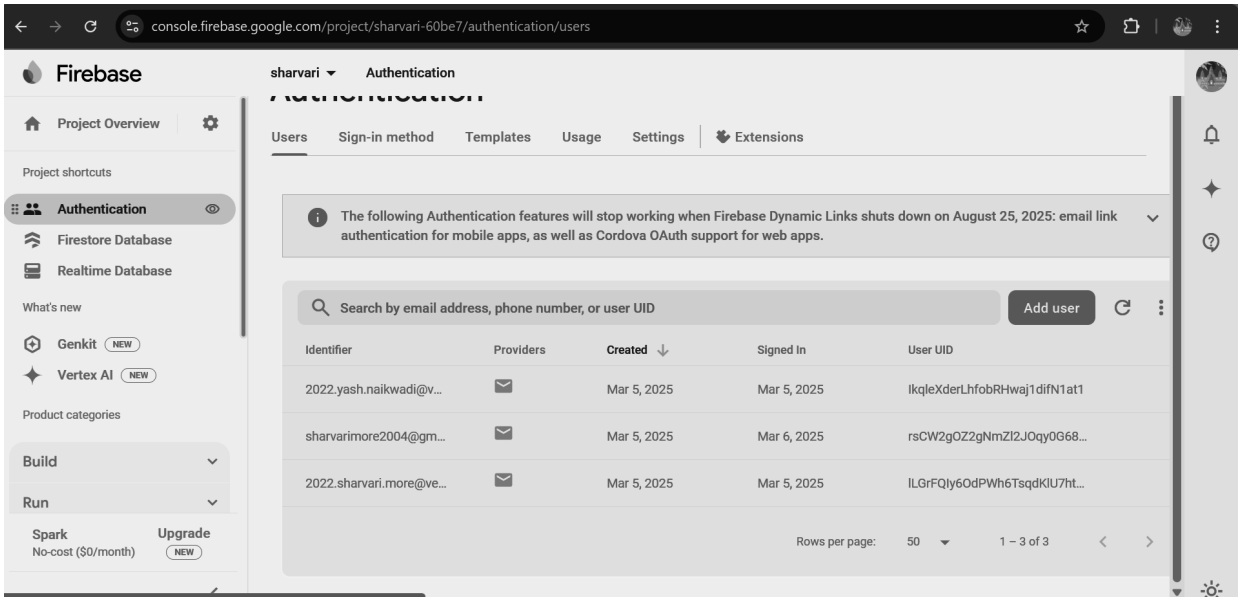

Reset your password for sharvari-804976081616
Inbox
noreply@sh... 7:18 AM
to me
Hello,
Follow this link to reset your sharvari-804976081616 password for your sharvarimore2004@gmail.com account.
https://sharvari-60be7.firebaseapp.com/__/auth/action?mode=resetPassword&oobCode=VtR3pnDDtClC8B1Yh5ecTNI0TVWeUlAcvgAffwbJ_tAAAAGVrBaueQ&apiKey=AIzaSyCjb1veZSX2FuehAOaHeTlxU_rDcx33ylg&lang=en
If you didn't ask to reset your password, you can ignore this email.
Thanks,
Your sharvari-804976081616 team


sharvari-60be7.fir...
sharvari-60be7.firebaseapp.com
Reset your password
for sharvarimore2004@gmail.com
New password
SAVE



Password changed

You can now sign in with your new password



Conclusion:

Integrating Firebase Authentication in our Flutter app enhanced security and user management. While implementing it, I encountered issues like delayed email verification updates and Firebase exceptions during sign-in. I resolved them by manually refreshing user authentication states and handling errors using try-catch blocks. Debugging authentication flows and ensuring proper dependency setup helped streamline the process, resulting in a smooth authentication system.