**LAPORAN PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK**



**JUDUL:**

**Authentikasi Firebase, Google, and Phone**

Disusun oleh:

M. Nur Aqil Bahri (21102144)

**TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO BANYUMAS, JAWA TENGAH**

**2024**

**Pembahasan Firebase**

Autentikasi Firebase adalah layanan penting bagi pengembang aplikasi mobile yang ingin membangun aplikasi yang aman dan mudah digunakan. Layanan ini menyediakan sistem login dan registrasi pengguna yang handal, dengan berbagai metode autentikasi yang fleksibel dan fitur-fitur canggih untuk meningkatkan keamanan dan pengalaman pengguna.

a. Manfaat Utama Autentikasi Firebase:

1. Meningkatkan Keamanan: Firebase Authentication menggunakan teknologi enkripsi dan otentikasi mutakhir untuk melindungi data pengguna dari akses tidak sah. Hal ini termasuk enkripsi kata sandi, verifikasi dua faktor, dan deteksi penipuan.
2. Mempermudah Integrasi: Firebase Authentication terintegrasi dengan mulus dengan Firebase lainnya, seperti Cloud Firestore dan Realtime Database, sehingga memudahkan pengelolaan data pengguna dan membangun aplikasi yang kompleks.
3. Meningkatkan Skalabilitas: Firebase Authentication dapat menangani aplikasi dengan jumlah pengguna yang besar secara efisien, tanpa perlu khawatir tentang infrastruktur server.
4. Menyediakan Fitur Lengkap: Firebase Authentication menawarkan berbagai fitur canggih, seperti manajemen sesi, pemulihan kata sandi, reset password, dan integrasi dengan penyedia identitas pihak ketiga.
5. Menghemat Waktu dan Biaya: Menggunakan Firebase Authentication dapat menghemat waktu dan biaya pengembangan yang terkait dengan membangun sistem

autentikasi kustom.

1. Mempermudah Pengelolaan Pengguna: Firebase Authentication menyediakan dashboard intuitif untuk mengelola akun pengguna, termasuk menambahkan, menghapus, dan mengubah informasi pengguna.
2. Mendukung Berbagai Platform: Firebase Authentication tersedia untuk berbagai platform pengembangan mobile populer, seperti Android, iOS, React Native, Flutter, dan Unity.
3. Memiliki Komunitas Besar: Firebase memiliki komunitas pengembang yang besar dan aktif, sehingga mudah untuk menemukan bantuan dan dukungan saat menggunakan layanan ini.

Autentikasi Firebase adalah solusi komprehensif dan fleksibel untuk kebutuhan autentikasi pengguna dalam aplikasi mobile. Layanan ini menawarkan kombinasi keamanan yang kuat, kemudahan penggunaan, skalabilitas, dan fitur-fitur canggih yang menjadikannya pilihan ideal bagi pengembang yang ingin membangun aplikasi mobile yang handal dan aman.

Dengan menggunakan Firebase Authentication, pengembang dapat fokus pada pengembangan fitur inti aplikasi mereka, tanpa perlu khawatir tentang kompleksitas sistem autentikasi. Hal ini memungkinkan mereka untuk menghadirkan aplikasi yang lebih aman, mudah digunakan, dan menarik bagi penggunanya.

**Langkah-Langkah Praktikum**

|  |  |
| --- | --- |
| **Langkah Praktikum** | **Pembahasan** |

|  |  |
| --- | --- |
| *Gambar 1. Halaman Awal Situs Firebase* | Membuat akun di *firebase.* |
| *Gambar 2. Install Firebase CLI* | Meng-*install firebase CLI.* Dengan mengunduh node.JS. |
| *Gambar 3. Installasi Node.JS* | *Install* node.js |
| *Gambar 4. Install CLI melalui npm* | Instal Firebase CLI melalui npm dengan menjalankan perintah berikut:  **npm install -g firebase-tools** |
| *Gambar 5. Tampilan membuat project firebase* | Membuat Project baru di firebase. |

|  |  |
| --- | --- |
| *Gambar 6. Tampilan setelah membuat project baru* | Menambahkan *flutter* ke *firebase.* |
| *Gambar 7. Perintah firebase login di command prompt* | Langkah pertama dalam menambahkan *flutter* ke *firebase* adalah dengan masuk ke *firebase* lewat *command prompt* dengan mengetikan perintah “*firebase*  *login”.* |
| *Gambar 8. login ke firebase CLI* | Setelah login ke *firebase* melalui *command prompt,* jika baru pertama kali melakukan, maka akan diarahkan untuk masuk ke akun yang terhubung dengan  *firebase CLI.* |
| *Gambar 9. Tampilan sukses terhubung dengan firebase CLI* | Setelah melakukan tahapan-tahapan sebelumnya, halaman sukses akan muncul seperti gambar di samping. |
| *Gambar 10. Membuat project flutter baru* | Membuat project flutter baru. |
| *Gambar 11. mengaktifkan paket flutterfire\_cli secara global* | menjalankan perintah “dart pub global activate flutterfire\_cli” membuat tool flutterfire dapat diakses dari mana saja di terminal |

|  |  |
| --- | --- |
| *Gambar 12. konfigurasi firebase ke flutter* | Perintah flutterfire configure -- project=prakflutterfirebase-38bb6 digunakan untuk mengkonfigurasikan firebase ke dalam project flutter. |
| *Gambar 13.konfigurasi berhasil* | Project berhasil dikonfigurasi. Dan berhasil mendapatkan firebase AppId |
| *Gambar 14. Menginstall dependensi* | Menginstall dependensi dengan menggunakan perintah Ketik “flutter pub add firebase\_core” |
| *Gambar 15. Mengubah void main* | Mengubah void main |
| *Gambar 16. Slicing UI* | Membuat kode untuk mengimplementasikan desain antarmuka pengguna pada gambar di samping. |
| *Gambar 17. file dalam folder ui* | Membuat folder baru yang diberi nama “ui”, dan di dalam folder tersebut buatlah 4 file dart sesuai pada gambar di samping. |

|  |  |
| --- | --- |
| *Gambar 18. kode untuk splash.dart* | Tambahkan kode di samping pada file splash.dart |
|  | Tambahkan kode di samping pada bagian file login.dart |

|  |  |
| --- | --- |
| *Gambar 19. kode untuk login.dart* |  |
|  | Tambahkan kode di samping pada bagian file register.dart |

|  |  |
| --- | --- |
| *Gambar 20. kode untuk register.dart* |  |
| *Gambar 21. kode untuk home\_screen.dart* | Tambahkan kode di samping pada bagian home\_screen.dart |
| *Gambar 22.file routes pada folder utils* | Selanjutnya kita perlu mendaftarkan state yang sudah kita buat dalam route dengan cara buat folder “utils” di dalam folder “lib”, dan buat file “routes.dart” di dalam folder”  utils”. |

|  |  |
| --- | --- |
| *Gambar 23. kode untuk routes.dart* | Tambahkan kode di samping pada bagian file routes.dart |
| *Gambar 24. kode untuk main.dart* | Tambahkan kode di samping pada bagian file main.dart |
| *Gambar 25. menambahkan authentikasi email di firebase* | Menambahkan authentikasi email dan password pada dashboard firebase di project kita. |
|  | Menginstal dependensi firebase\_auth |

|  |  |
| --- | --- |
| *Gambar 26. dependensi firebase\_auth* |  |

|  |  |
| --- | --- |
| *Gambar 27. cek firebase\_auth* | Cek ketersediaan dependensi firebase\_auth pada pubspec.yaml |
| *Gambar 28. menambahkan dependensi lain* | Untuk memproses logika bisnis pada aplikasi kita akan menggunakan flutter bloc,  sehingga perlu menambahkan 2 dependensi yaitu :  bloc: ^8.1.2 flutter\_bloc: ^8.1.3  jangan lupa untuk menjalankan perintah  “flutter pub upgrade” |
| *Gambar 29. menambahkan file baru pada folder baru repositories* | Buat folder “repositories” pada folder “lib”, kemudian buat file “auth\_repo.dart”.  folder repositories ini akan kita gunakan untuk berkomunikasi dengan firebase/API dari back  end. |
| *Gambar 30. kode untuk file auth\_repo.dart* | Kita akan membuat 2 buah Future<void> dalam class AuthRepo untuk login dan register pada file auth\_repo.dart. |

|  |  |
| --- | --- |
| *Gambar 31. struktur untuk mengatur bloc* | Kemudian kita akan mengatur bloc untuk login. Buat folder dengan struktur seperti gambar di samping. |
| *Gambar 32. kode untuk file login\_state.dart* | Tambahkan kode di samping pada file login\_state.dart |
| *Gambar 33. kode untuk file login\_cubit.dart* | Tambahakan kode di samping pada file login\_cubit.dart |

|  |  |
| --- | --- |
| *Gambar 34. Run program* | Jalankan program. inputan email dan password akan disimpan pada emailEdc dan passEdc kemudian  dengan menggunakan bloc inputan tersebut dikirim ke firebase melalui repositori yang sudah  kita buat, apabila berhasil maka akan masuk ke home screen, apabila gagal akan muncul  pesan gagal. |
| *Gambar 35. menambahkan user* | Mengatur untuk bagian register di firebase |

|  |  |
| --- | --- |
| *Gambar 36. berhasil login* | Data yang baru ditambahkan, kita cek untuk mengisi pada bagian login. |
| *Gambar 37. folder register* | Login sudah berhasil kita buat, maka selanjutnya kita akan membuat bagian register.  Karena kita sudah membuat Future<void> register. Selanjutnya kita atur untuk blocnya. Buat  folder “register” di dalam folder bloc. |

|  |  |
| --- | --- |
| *Gambar 38. kode file register\_state.dart* | Tambahkan kode di samping pada bagian file register\_state.dart |
| *Gambar 39. kode file register\_cubit.dart* | Tambahkan kode di samping pada bagian file register\_cubit.dart |
|  | Menggunakan bloc pada register.dart,  sehingga kode program menjadi seperti berikut |
|  |  |

|  |  |
| --- | --- |
| *Gambar 40. test register* | Mencoba register |
| *Gambar 41. data akun yang masuk* | User yang baru didaftarkan berhasil ditambahkan |

|  |  |
| --- | --- |
| *Gambar 42. bukti login berhasil* | Mencoba login dengan akun yang baru saja di tambahkan |
| *Gambar 43. kode tambahan untuk login.dart* | Menambahkan authentikasi baru menggunakan Google dan Phone, dengan menambahkan beberapa fungsi. Yang pertama tambahkan icon google dan Phone pada kode login.dart |

|  |  |
| --- | --- |
|  | Icon berhasil ditambahkan |
| *Gambar 44. menambahkan depedensi* | Tambahkan dependency google\_sign\_in dengan menjalankan command flutter pub add google\_sign\_in pada terminal. |
| *Gambar 45. menambah metode masuk* | Untuk mengaktifkan otentikasi menggunakan akun Google perlu menambahkan sign\_in method Google pada web firebase |
| *Gambar 46. menambahkan fingerprint pada project* | Untuk menggunakan otentikasi menggunakan akun Google pada Android perlu  menambahkan fingerprint pada Project Settings web firebase. Dengan cara pergi pe project settings dan pilih fingerprints |

|  |  |
| --- | --- |
| *Gambar 47. infromasi SHA* | Untuk mendapatkan SHA1 dan SHA256 menggunakan command berikut pada Terminal: keytool -list -v \  -alias androiddebugkey - keystore %USERPROFILE%\.android\deb ug.keystore  kemudian tambahkan kode SHA1 ke  firebase |
| *Gambar 48. google service* | Unduh Google service.json |
| *Gambar 49. file Google Service.json yang baru* | Ganti file google service yang lama dengan yang baru pada direktori app |
| *Gambar 50. kode tambahan pada login.dart* | Masih pada login.dart tambahkan fungsi berikut untuk sign in menggunakan akun Google |
| *Gambar 51. kode tambahan pada login.dart* | Tambahkan fungsi signInWithGoogle pada logo Google |

|  |  |
| --- | --- |
| *Gambar 52. signin Google* | Disini login menggunakan Google sudah bisa diakses, |
|  | Menambahkan tombol keluar pada home\_screen |

|  |  |
| --- | --- |
|  | Tampilan home\_screen terbaru |
|  | Menambahkan fungsi keluar pada tombol keluar |
|  | Menambahkan metode login baru yaitu dengan menggunakan phone |
|  | Tambahkan fungsi route ke phone\_auth\_screen |

|  |  |
| --- | --- |
|  | Pada folder ui, tambahkan file phone\_auth\_screen.dart |
|  | Tambahkan kode di samping pada file phone\_auth\_screen,dart |
|  | Tampilan login menggunakan phone |

|  |  |
| --- | --- |
|  | Menambahkan TextEditingController dan Dispose controller tersebut agar tidak terjadi memory leak |
|  | Pasang phone controller pada text field |
|  | Buat fungsi untuk otentikasi menggunakan nomor handphone. |
|  | Buat fungsi OTP |

|  |  |
| --- | --- |
|  | Memasukan nomor telepon |