**LAPORAN PRAKTIKUM**

|  |  |  |
| --- | --- | --- |
| **IDENTITAS MAHASISWA** | | |
| **Nama Mahasiswa** | : | Firza Aji Karya |
| **NPM** | : | 223307007 |
| **Kelas** | : | 3A |

|  |
| --- |
| **DASAR TEORI** |
| Dalam GitHub, repository adalah tempat menyimpan file proyek yang dapat berupa file source code, file gambar, text, dan file lainnya. Umumnya repository memang digunakan untuk menyimpan source code karena berhubungan dengan pemrograman.  Arsitektur aplikasi adalah seperangkat aturan desain. Mirip dengan cetak biru sebuah rumah, arsitektur memberikan struktur bagi aplikasi Anda. Arsitektur aplikasi yang baik dapat membuat kode Anda andal, fleksibel, skalabel, dan mudah dikelola selama bertahun-tahun. mempelajari cara menggunakan ViewModel, salah satu komponen Arsitektur untuk menyimpan data aplikasi Anda. Data yang disimpan tidak akan hilang jika framework menghancurkan dan membuat ulang aktivitas dan fragmen selama perubahan konfigurasi atau peristiwa lainnya. |

|  |
| --- |
| **ALAT DAN BAHAN** |
| 1. Laptop 2. Android Studio 3. GitHub |

|  |
| --- |
| **KODE PROGRAM DAN HASIL** |
| Membuat Repository   * Klik link berikut <https://github.com/google-developer-training/android-basics-kotlin-unscramble-app> * Pilih segitiga terbalik pada fork , lalu klik tanda **+ Create a new fork**      * Ganti nama Repository, jangan centang pada **copy the main branch only** lalu **create fork**      * Repository sudah berhasil dibuat, kemudian **Clone** dengan cara klik **Code** lalu copy link **SSH**      * Masuk ke Android Studio, pilih File -> New -> Project from Version Control      * Paste link SSH yang sudah di copy, kemudian **Clone**      * Invite teman ke Repository yang sudah dibuat dengan cara masuk ke Setting -> Collaborators -> Add People -> Masukkan Username GitHub teman , tunggu teman menerima invite      * Buat branch yang awalnya main menjadi **original/starter** lalu **Checkout**     CodeLab  4. Menambahkan View Model   * Buat file class Kotlin baru bernama **GameViewModel,** Di jendela **Android**, klik kanan pada folder **ui.game**. Pilih **New** > **Kotlin File/Class**.      * Ubah **GameViewModel** menjadi subclass dari **ViewModel**      * Pada bagian atas class **GameFragment** tambahkan properti jenis **GameViewModel .** Lakukan inisialisasi menggunakan properti Kotlin **by viewModels()**     5. Memindahkan data ke ViewModel   * Pindahkan variabel data score, currentWordCount, currentScrambleWord ke class **GameViewModel**      * Dalam GameViewModel,Ubah deklarasi currentScrambleWord untuk menambahkan properti pendukung * Di Game Fragment, perbarui metode updateNextWordOnScreen()     6. Memahami siklus proses ViewModel   * Menambahkan blok init di **GameViewModel.kt**      * Menambahkan laporan log didalam onCleared()      * Di GameFragment dalam onCreateView(), setelah Anda mendapatkan referensi ke objek binding, tambahkan laporan log untuk mencatat pembuatan fragmen.      * Di GameFragment, ganti metode callback onDetach(), yang akan dipanggil saat aktivitas dan fragmen yang terkait dihancurkan.     7. Mengisi ViewModel   * Pada GameViewModel, tambahkan variabel class baru jenis MutableList<String> yang disebut wordsList. Tambahkan variabel class lain yang disebut currentWord untuk menyimpan kata yang ingin disusun oleh pemain. Gunakan kata kunci lateinit karena Anda akan menginisialisasi properti ini nanti.      * Tambahkan metode private baru yang disebut getNextWord(), di atas blok init, tanpa parameter yang tidak menampilkan apa-apa.Dapatkan kata acak dari allWordsList dan tetapkan ke currentWord.      * Di getNextWord(), konversikan string currentWord ke array karakter dan tetapkan ke val baru yang disebut tempWord.      * Terkadang urutan karakter yang diacak sama dengan kata aslinya. Tambahkan loop while berikut di sekitar panggilan untuk mengacak, untuk melanjutkan loop hingga kata yang diacak tidak sama dengan kata asli.      * Tambahkan blok if-else untuk memeriksa apakah suatu kata telah digunakan. Jika wordsList berisi currentWord, panggil getNextWord().      * Untuk menampilkan kata yang ejaannya diacak di awal aplikasi, Anda perlu memanggil metode getNextWord(), yang selanjutnya memperbarui currentScrambledWord. Lakukan panggilan ke metode getNextWord() di dalam blok init pada GameViewModel.      * Tambahkan pengubah lateinit ke properti \_currentScrambledWord. Tambahkan penyebutan eksplisit jenis data String,      * Hasil     8. Dialog   * tambahkan properti pendukung ke variabel score. Di GameViewModel      * Di GameFragment, tambahkan fungsi pribadi bernama showFinalScoreDialog(). Untuk membuat MaterialAlertDialog, gunakan class MaterialAlertDialogBuilder untuk membuat bagian dialog langkah demi langkah.      * Tambahkan kode untuk menetapkan judul pada dialog pemberitahuan, gunakan resource string      * Setel pesan agar menampilkan skor akhir, gunakan versi hanya baca dari variabel skor (viewModel.score), yang telah Anda tambahkan sebelumnya.      * Jadikan dialog pemberitahuan tidak dapat dibatalkan saat tombol kembali ditekan, menggunakan metode [setCancelable()](https://developer.android.com/reference/android/app/Dialog?hl=id" \l "setCancelable(boolean)" \t "_blank) dan meneruskan false.      * Tambahkan dua tombol teks **EXIT** dan **PLAY AGAIN** menggunakan metode setNegativeButton() dan setPositiveButton(). Panggil masing-masing exitGame() dan restartGame() dari lambda.      * Di bagian akhir, tambahkan [show()](https://developer.android.com/reference/android/app/Dialog?hl=id#show()), yang akan membuat lalu menampilkan dialog pemberitahuan.     9. mengimplementasikan OnClickListener untuk tombol Submit   * Hapus kode di dalam onSubmitWord() yang akan dipanggil saat tombol **Submit** diketuk. * Tambahkan centang pada nilai return metode viewModel.nextWord(). Jika true, kata lain tersedia, jadi perbarui kata yang ejaannya diacak di layar menggunakan updateNextWordOnScreen().      * Mencoba      * Di GameViewModel, tambahkan metode pribadi baru yang disebut increaseScore() tanpa parameter dan nilai yang ditampilkan. Tingkatkan variabel score sebesar *SCORE\_INCREASE*.      * Dalam GameViewModel, tambahkan metode bantuan yang disebut isUserWordCorrect() yang menampilkan Boolean dan mengambil String, kata dari pemain, sebagai parameter. * Di isUserWordCorrect() validasikan kata dari pemain dan tingkatkan skor jika tebakannya benar. Tindakan ini akan memperbarui skor akhir dalam dialog pemberitahuan.      * Pada GameFragment, di awal onSubmitWord(), buat val yang bernama playerWord      * Pada onSubmitWord(), di bawah deklarasi playerWord, validasikan kata pemain. Tambahkan pernyataan if untuk memeriksa kata pemain menggunakan metode isUserWordCorrect(), dengan meneruskan playerWord. * Di dalam blok if, reset kolom teks, panggil setErrorTextField dengan memasukkan false. * Pindahkan kode yang ada ke dalam blok if.      * Jika kata pengguna salah, tampilkan pesan error di kolom teks. Tambahkan blok else ke blok if di atas, lalu panggil setErrorTextField() yang meneruskan true.      * Jalankan Aplikasi     10. Mengimplementasikan tombol Skip   * Serupa dengan onSubmitWord(), tambahkan kondisi dalam metode onSkipWord(). Jika true, tampilkan kata di layar dan reset kolom teks. Jika false dan tidak ada lagi kata yang tersisa di babak ini, tampilkan dialog pemberitahuan dengan skor akhir.     11. Memverifikasi ViewModel mempertahankan data   * Untuk tugas ini, tambahkan logging di GameFragment untuk mengamati bahwa data aplikasi Anda disimpan di ViewModel, selama perubahan konfigurasi.   12. Memperbarui logika restart game   * Untuk mereset data aplikasi, di GameViewModel, tambahkan metode yang disebut reinitializeData(). Tetapkan skor dan jumlah kata menjadi 0. Hapus daftar kata dan panggil metode getNextWord().      * Pada GameFragment di bagian atas metode restartGame(), lakukan panggilan ke metode yang baru dibuat, reinitializeData().      * Jalankan |

|  |
| --- |
| **KESIMPULAN** |
| Pada praktikum ini kita dapat mengetahui bagaimana cara membuat Repository dan menginvite teman di GitHub dan mengClone di Android Studio. Kita juga dapat memperlajari codelab tentang arsitektur aplikasi bagaimana cara menambahkan ViewModel, Memindahkan data ke ViewModel, dan mengisi ViewModel. |