Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi
https://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/index
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0893

Pengembangan Fitur Lanjutan Pada Aplikasi Manajemen Personal Individual Berbasis *Mobile* Android

DOI: http://dx.doi.org/10.35889/jutisi.v14i3.3168

Billy Prestone Mahulae[1*], Yohana Tri Widayati[2], Satrio Agung Prakoso[3]
Teknik Informatika, Universitas AKI Semarang, Semarang, Indonesia
*e-mail *Corresponding Author:* billyprestonem@gmail.com

Abstract

The rapid development of technology and personal management challenges among Indonesian youth require an integrated solution to manage both finances and time. This study develops advanced features on the Individual Personal Management Application (AMPI) for Android using the waterfall method, clean MVVM architecture, Coin dependency injection, and Firebase integration for authentication and data storage. A user needs analysis through an online questionnaire with 50 respondents showed a high interest in an integrated application with reminders, data visualization, task prioritization, and secure data storage. The system is designed using use case, activity, sequence, and class diagrams to support account, financial, and time management. Black-box testing confirmed that all features function as designed. The results demonstrate that AMPI effectively assists users, especially young people, in tracking daily transactions, managing schedules, and monitoring productivity and financial conditions.

Keywords: *Personal Management Application; Android; Financial Management; Time Management; Firebase*

Abstrak

Perkembangan teknologi dan tantangan manajemen personal pada generasi muda Indonesia menuntut solusi aplikasi yang dapat mengelola keuangan dan waktu secara terintegrasi. Penelitian ini mengembangkan fitur lanjutan pada *Aplikasi Manajemen Personal Individual (AMPI)* berbasis Android menggunakan metode *waterfall*, arsitektur MVVM clean, Koin dependen injeksi, serta integrasi Firebase untuk autentikasi dan penyimpanan data. Analisis kebutuhan pengguna melalui kuesioner daring pada 50 responden menunjukkan minat tinggi terhadap aplikasi ter-integrasi dengan fitur mengingat, visualisasi data, prioritas tugas, dan penyimpanan data aman. Sistem dirancang menggunakan *use case*, aktivitas, *sequence*, dan *class diagram* untuk mendukung manajemen akun, keuangan, dan waktu. Pengujian black-box menunjukkan seluruh fitur berfungsi sesuai rancangan. Hasil penelitian menunjukkan AMPI efektif membantu pengguna, khususnya generasi muda, dalam mencatat transaksi harian, mengatur jadwal, serta memantau produktivitas dan kondisi keuangan.

Kata kunci: *Aplikasi Manajemen Personal; Android; Manajemen Keuangan; Manajemen Waktu; Firebase*

1. Pendahuluan

Dalam era digital yang berkembang pesat, kemampuan mengelola keuangan pribadi dan waktu menjadi keterampilan penting bagi generasi muda. Gaya hidup modern yang dinamis, ditambah dengan arus informasi dan teknologi yang bergerak cepat, menuntut anak muda untuk memiliki manajemen personal yang baik agar aktivitas sehari-hari tetap produktif, teratur, dan seimbang.

Namun, berbagai studi menunjukkan bahwa mayoritas generasi muda di Indonesia belum memiliki literasi finansial dan manajemen waktu yang memadai. Laporan OCBC NISP *Financial Fitness Index* menunjukkan bahwa 85% anak muda Indonesia berada dalam kondisi finansial tidak sehat, dengan skor literasi keuangan rata-rata 37,72 dari 100, yang mencerminkan

rendahnya pemahaman dalam mengelola keuangan. Selain itu, penelitian terkait kebiasaan belajar remaja menemukan bahwa banyak anak muda mengalami kesulitan menetapkan prioritas, kurang memiliki jadwal teratur, dan cenderung menunda tugas, sehingga berdampak pada rendahnya produktivitas dan sulitnya pencapaian tujuan jangka panjang. Kondisi ini memperlihatkan adanya masalah nyata dan terukur dalam pengelolaan keuangan dan waktu pada kalangan generasi muda di Indonesia [1].

Pemanfaatan teknologi, khususnya perangkat mobile berbasis Android, berpotensi menjadi solusi untuk membantu meningkatkan kemampuan manajemen personal. Berbagai penelitian menyebutkan bahwa perangkat mobile kini tidak hanya menjadi alat komunikasi, tetapi juga digunakan untuk mengatur aspek-aspek kehidupan seperti jadwal, catatan, tugas, dan keuangan [2].

Namun, aplikasi yang tersedia umumnya hanya berfokus pada satu aspek tertentu, sehingga pengguna harus menggunakan banyak aplikasi sekaligus [3]. Oleh karena itu, pengembangan aplikasi manajemen personal berbasis Android yang mengintegrasikan fitur manajemen keuangan dan manajemen waktu secara komprehensif dipandang sebagai solusi yang efektif [4].

Integrasi ini memungkinkan pengguna mencatat pemasukan–pengeluaran, merencanakan anggaran, mengatur jadwal harian, hingga mengelola tugas dalam satu platform, sehingga lebih efisien dan mudah digunakan oleh generasi muda [5] .

Penelitian ini bertujuan untuk mengembangkan aplikasi manajemen personal berbasis Android yang menggabungkan fitur manajemen keuangan dan waktu dalam satu sistem terintegrasi. Adapun manfaat penelitian ini adalah membantu meningkatkan literasi finansial, kemandirian ekonomi, kedisiplinan, serta produktivitas generasi muda. Dengan adanya aplikasi yang dirancang sesuai kebutuhan pengguna muda, diharapkan mereka mampu membentuk kebiasaan hidup yang lebih teratur, bijak secara finansial, dan lebih siap menghadapi tantangan kehidupan modern [6].

2. Tinjauan Pustaka

Pada penelitian terdahulu telah [7] berjudul "Desain UI/UX Aplikasi Manajemen Keuangan Pribadi Menggunakan Metode *User Centered Design* (UCD)" menghasilkan sebuah aplikasi pengelola keuangan pribadi berbasis Android. Teknologi yang digunakan dalam penelitian ini antara lain Kotlin, Firebase, dan Android Studio. “Putri dan Fadillah [8] menjelaskan bahwa integrasi Firebase *Authentication* dan *Realtime* Database mampu meningkatkan keamanan data pengguna serta mempermudah proses autentikasi pada aplikasi berbasis Android. Hal ini menunjukkan bahwa penggunaan Firebase menjadi solusi efektif dalam pengembangan aplikasi yang membutuhkan penyimpanan data dan autentikasi yang andal.” Aplikasi yang dikembangkan mampu membantu pengguna dalam mengontrol pengeluaran, menghindari pemborosan, serta mengatur keuangan berdasarkan kategori kebutuhan seperti primer, sekunder, dan tersier. Pendekatan UCD digunakan agar sistem benar-benar sesuai dengan kebutuhan pengguna.

Penelitian serupa juga dilakukan oleh [9] yang mengembangkan "Pengembangan Sistem Saran Keuangan Untuk Mahasiswa (Anak Kos) Berbasis Mobile Android". 10 Teknologi yang digunakan meliputi Kotlin, Android Studio, dan MySQL. Sistem yang dihasilkan mampu memberikan saran keuangan kepada mahasiswa dan telah terintegrasi langsung dengan database serta server publik. Pengembangan aplikasi ini menggunakan pendekatan *User Centered Design* (UCD) dan menerapkan arsitektur Model View Presenter (MVP) guna meningkatkan efektivitas sistem dalam memberikan rekomendasi pengelolaan keuangan.

Selain itu, penelitian yang dilakukan oleh Ardiansyah dkk. [10] membahas implementasi arsitektur MVVM pada aplikasi manajemen keuangan berbasis Android yang terbukti meningkatkan modularitas dan keteraturan kode. Prasetya dan Aditya [11] juga menunjukkan bahwa penerapan *clean architecture* mampu memperbaiki struktur kode aplikasi keuangan pribadi agar lebih mudah di-maintain dan dikembangkan.

Sementara itu, [12] merancang dan membangun sistem manajemen keuangan pribadi menggunakan model budget jar berbasis Android. Teknologi yang digunakan mencakup Kotlin, Java, Android Studio, dan SQLite. Hasil penelitian menunjukkan bahwa sistem yang dikembangkan berhasil mengimplementasikan konsep pengelolaan anggaran budget jar secara efektif, di mana sistem mampu mengalokasikan anggaran berdasarkan kategori toples, memberikan peringatan saat pengeluaran melebihi batas anggaran, serta menyajikan laporan mutasi anggaran secara informatif.

Penelitian lain yang relevan dilakukan oleh [13] dengan judul "Pengembangan Aplikasi Mobile HabitTroops". Penelitian ini memanfaatkan teknologi Kotlin, Android Studio, dan Firebase. Aplikasi HabiTroops dikembangkan dengan antarmuka yang intuitif dan berhasil membantu pengguna dalam menjalankan aktivitas kebiasaan secara teratur. Fokus utama dari aplikasi ini adalah pembentukan kebiasaan positif dengan bantuan fitur-fitur mengingat dan pelacakan kegiatan harian.

"Sementara itu, Nugroho dkk. [14] mengembangkan aplikasi manajemen waktu dan produktivitas menggunakan metode waterfall yang dilengkapi fitur pengingat tugas harian. Hasil penelitian mereka menunjukkan peningkatan efisiensi dan kedisiplinan pengguna dalam mengelola jadwal."

Penelitian ini menghadirkan kebaruan dengan menggabungkan beberapa aspek yang belum terintegrasi secara simultan pada penelitian terdahulu: (1) integrasi penuh manajemen keuangan dan manajemen waktu dalam satu aplikasi Android sedangkan studi prior menekankan salah satu aspek saja membuat pengguna tidak perlu berpindah-pindah aplikasi untuk mengelola anggaran dan jadwal; (2) dari sisi arsitektur dan teknik, aplikasi ini mengimplementasikan Clean MVVM dipadukan Koin (DI) dan Firebase (*Authentication + Realtime Database*) sehingga meningkatkan modularitas, testabilitas, dan keamanan data dibandingkan implementasi MVP/MVVM sederhana atau arsitektur monolitik pada studi sebelumnya; (3) secara fungsional, AMPI menambahkan fitur lanjutan yang jarang hadir bersamaan pada literatur seperti notifikasi pengingat, visualisasi data interaktif (grafik donut), pengelompokan tugas berdasarkan prioritas, dan rangkuman ringkas produktivitas/keuangan yang dirancang berdasarkan analisis kebutuhan pengguna (kuesioner 50 responden); serta (4) dari sisi validasi, pengujian fungsional dilakukan melalui *black box* dengan skenario lengkap pada modul akun, keuangan, dan waktu, memberikan bukti operasionalitas *end-to-end*. Kombinasi integrasi domain, arsitektur modern, fitur fungsional lengkap, dan pendekatan evaluasi inilah yang menjadikan kontribusi penelitian ini bersifat inovatif dan aplikatif dibandingkan karya terdahulu.

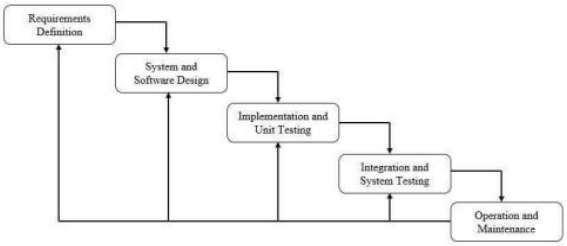
3. Metodologi

3.1. Gambaran Umum Aplikasi Versi Awal AMPI

Aplikasi Manajemen Personal Individual (AMPI) merupakan sebuah aplikasi berbasis android yang dirancang untuk membantu pengguna dalam melakukan manajemen pribadi seperti manajemen keuangan dan manajemen waktu. Pada aplikasi versi awal dirancang dengan menggunakan bahasa pemrograman kotlin serta menggunakan room database sebagai penyimpanan data lokal [15]. Fungsionalitas utama pada aplikasi versi awal berfokus pada operasi dasar CRUD (*Create, Read, Update, Delete*) yang memungkinkan pengguna dapat mencatat, melihat, memperbarui dan menghapus data pada manajemen keuangan dan manajemen waktu secara mandiri. Pada aplikasi AMPI versi awal telah mampu memenuhi kebutuhan dasar terkait pencatatan dan pelacakan, namun aplikasi versi awal masih memiliki keterbatasan, terutama karena fitur yang tersedia masih bersifat parsial serta antarmuka yag sederhana dan kurang interaktif.

3.2. Metode Pengembangan

Metode yang digunakan dalam Pengembangan Fitur Lanjutan Pada Aplikasi Manajemen Personal Individual (AMPI) Berbasis Android adalah metode *waterfall*. Metode *waterfall* yang digunakan dapat dilihat pada gambar.



Gambar 1 Tahapan Metode *Waterfall*[16]

Penggunaan metode *waterfall* karena pengaplikasiannya yang mudah serta proses metode tersebut sudah menjangkau seluruh proses pengembangan aplikasi. Kelebihan lain yakni

Ketika kebutuhan pengembangan system dirancang dengan baik dan benar maka metode *waterfall* akan berjalan dengan baik tanpa masalah**.** Pada sub bab ini akan menyajikan sebuah data hasil analisis kebutuhan pengguna dan kebutuhan aplikasi dalam pengembangan fitur lanjutan Aplikasi Manajemen Personal Individual (AMPI).

1) Kebutuhan Pengguna

Hasil survei menunjukkan mayoritas responden (60%) berusia 21–23 tahun dan berada pada tahap pendidikan atau awal karier. Sebagian besar responden (76%) berusaha membuat jadwal harian, namun 56% menyatakan manajemen waktu mereka hanya “cukup terkelola”, dengan penyebab utama penundaan tugas adalah rasa malas (56%). Pada aspek keuangan, 50% hanya mencatat pemasukan dan pengeluaran “saat ingat saja”, dan 42% responden sering membeli barang yang tidak penting sehingga 40% di antaranya kerap kehabisan uang sebelum akhir bulan. Minat terhadap aplikasi terintegrasi sangat tinggi, dengan 86% responden menyatakan tertarik, serta fitur paling dibutuhkan meliputi notifikasi pengingat (100%), visualisasi grafik (100%), pengelompokan prioritas tugas (96%), dan penyimpanan data aman (86%). Data ini menegaskan perlunya aplikasi manajemen personal terintegrasi untuk membantu pengguna meningkatkan kedisiplinan waktu dan pengelolaan keuangan.

2) Kebutuhan Aplikasi

1). Kebutuhan Fungsional

a) Manajemen Akun
   - Pengguna dapat membuat akun baru.
   - Pengguna dapat melakukan *login* dan *logout.*
   - Pengguna dapat mereset kata sandi melalui email.
   - Sistem menyimpan dan memverifikasi data akun melalui *Firebase Authentication*.

b) Manajemen Waktu
   - Pengguna dapat menambahkan, mengedit, dan menghapus tugas.
   - Pengguna dapat mengelompokkan tugas berdasarkan prioritas (tinggi, sedang, rendah).
   - Sistem menampilkan daftar tugas yang terorganisasi berdasarkan tanggal dan prioritas.
   - Sistem memberikan notifikasi pengingat tugas sesuai waktu yang ditentukan pengguna.

c) Manajemen Keuangan
   - Pengguna dapat mencatat pemasukan.
   - Pengguna dapat mencatat pengeluaran.
   - Pengguna dapat mengedit dan menghapus transaksi.
   - Pengguna dapat mengelompokkan transaksi berdasarkan kategori (misal: kebutuhan, keinginan, lainnya).
   - Sistem menyajikan visualisasi grafik pemasukan–pengeluaran.
   - Sistem menampilkan ringkasan keuangan harian, mingguan, atau bulanan.

d) Penyimpanan dan Sinkronisasi Data
   - Semua data tugas dan transaksi tersimpan otomatis ke Firebase Realtime Database.
   - Data pengguna tersinkronisasi secara real-time antar perangkat.

e) Dashboard Personal
   - Sistem menampilkan dashboard berisi ringkasan waktu (task progress) dan keuangan (total pemasukan, total pengeluaran, dan grafik).

2) Kebutuhan Nonfungsional

a) Keamanan
   - Autentikasi pengguna menggunakan Firebase Authentication
   - Data tersimpan aman pada Firebase Realtime Database dengan aturan akses (rules) terproteksi.

b) Kinerja Sistem
   - Aplikasi harus memuat data dalam waktu kurang dari 3 detik.
   - Proses sinkronisasi data berlangsung real-time tanpa penundaan signifikan.

c) Usability
   - Tampilan antarmuka sederhana, konsisten, dan mudah digunakan.
   - Navigasi aplikasi dapat dipahami pengguna tanpa pelatihan khusus.

d) Reliabilitas
   - Aplikasi mampu berjalan tanpa crash dalam penggunaan normal.

- Sistem tetap berfungsi meskipun koneksi internet tidak stabil (menyimpan data sementara dan mengunggah ketika online).

e) Kompatibilitas
- Aplikasi dapat berjalan pada perangkat Android minimal versi 8 (Oreo) atau lebih baru.

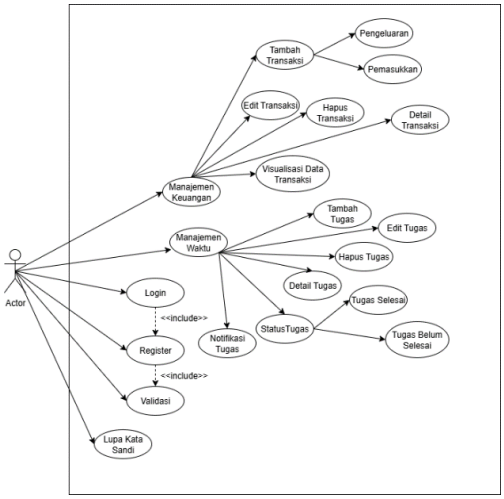
f) Maintainability
- Arsitektur Clean MVVM diterapkan agar aplikasi mudah dikembangkan dan diperbaiki.

g) Portabilitas
- Aplikasi dapat diinstal dan dijalankan pada berbagai ukuran layar dan perangkat Android.
-

3.3. Perancangan Sistem

1) *Use Case* Diagram

Berikut merupakan sebuah desain sistem *use case* diagram pada Aplikasi Manajemen Personal Individual (AMPI).

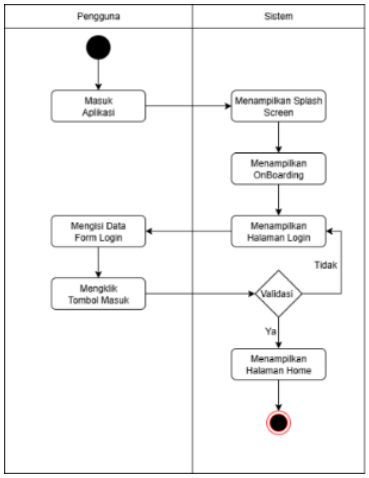


Gambar 2 Rancangan Use Case Diagram

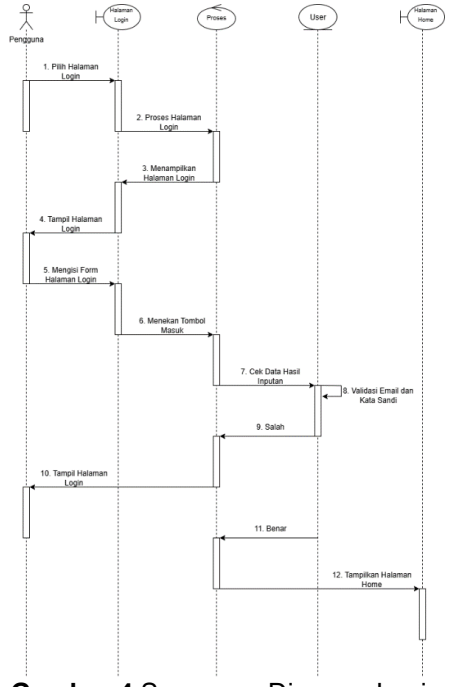
Pada *use case* di atas terdapat beberapa bagian yang telah ditentukan dalam tahapan analisis kebutuhan seperti manajemen akun (login*, register*, validasi, dan lupa kata sandi), manajemen keuangan (CRUD transaksi, detail transaksi, dan visualisasi data), serta manajemen waktu (CRUD tugas, status tugas, dan notifikasi tugas).

2) *Activity* Diagram Dan *Sequence* Diagram

Berikut merupakan sebuah rancangan *activity* diagram dan *sequence* diagram pada fitur login.



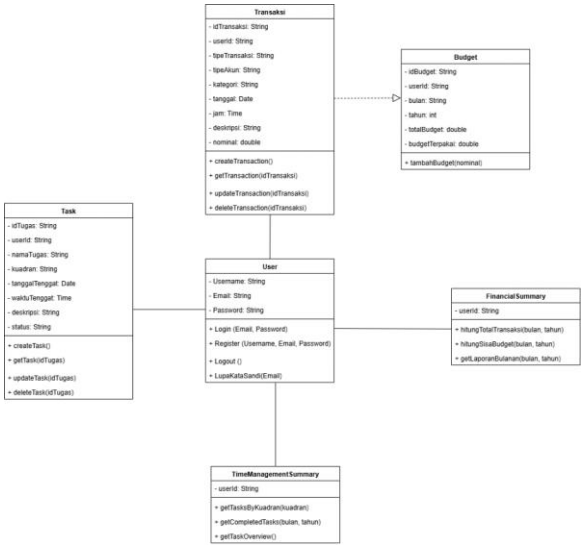
Gambar 3 *Activity* Diagram Login



Gambar 4 S*equence* Diagram Login

Aktivitas diagram login menggambarkan alur dari awal aplikasi hingga masuk ke halaman utama atau kembali ke login jika validasi gagal, sedangkan *sequence* diagram menampilkan urutan interaksi antar objek pengguna, halaman login, proses, dan objek user dalam verifikasi kredensial.

3) *Class* Diagram



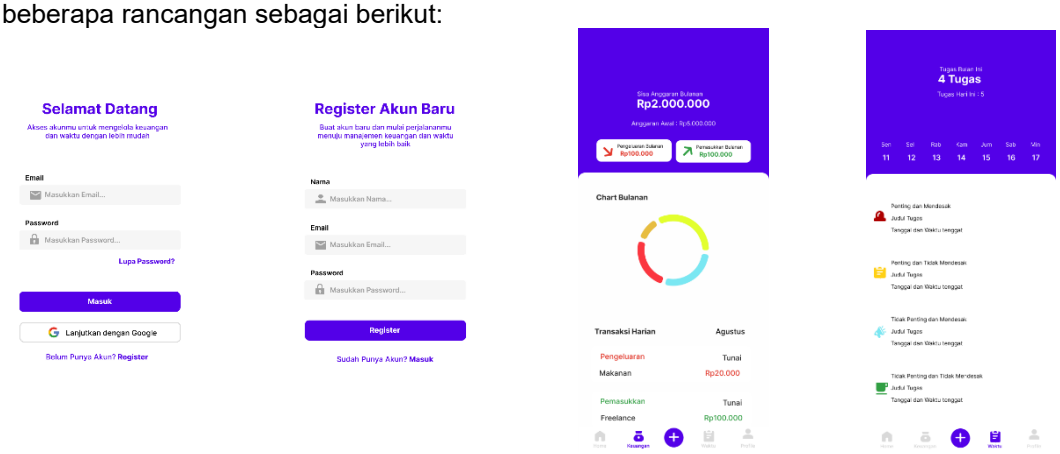
Gambar 5 Rancangan *Class* Diagram

Class diagram menggambarkan arsitektur sistem manajemen personal ter-integrasi dengan enam kelas utama. Kelas *User* sebagai inti sistem menangani otentikasi dan identitas pengguna, serta memiliki banyak Task dan Transaksi. Task menyimpan detail tugas harian dengan metode CRUD, sedangkan Transaksi mencatat aktivitas keuangan lengkap dengan metode CRUD. Kelas Budget mengelola anggaran bulanan terkait Transaksi, dan kelas *TimeManagementSummary* serta *FinancialSummary* menyediakan laporan produktivitas dan kondisi keuangan secara terpusat, masing-masing terkait satu-per-satu dengan User.

4. Hasil dan Pembahasan

4.1 Implementasi Sistem

Tahap selanjutnya adalah penerapan desain interface untuk implementasi aplikasi, dengan beberapa rancangan sebagai berikut:



Gambar 6 Desain Interface

4.1. Implementasi Teknis

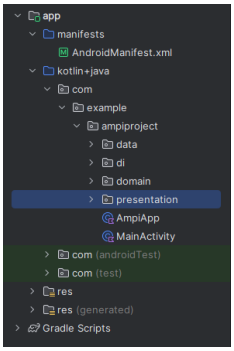
1) Implementasi Clean Architecture MVVM dengan Koin

Tahap pertama pengembangan fitur lanjutan AMPI adalah menggunakan library Koin sebagai Dependency Injection (DI) untuk mengelola dependensi tanpa kode boilerplate berlebih.

```
//Library Koin
implementation(libs.koin.android)
implementation (libs.koin.androidx.compose)
implementation (libs.koin.androidx.compose.navigation)
```

Gambar 7 Implementasi Library Koin

Setelah menambahkan library Koin, langkah berikutnya adalah membuat struktur folder kode aplikasi untuk merapikan dan mengkategorikan setiap bagian.



Gambar 8 Struktur Folder Clean Architecture

2) Implementasi Firebase

Pengembangan aplikasi ini menggunakan Firebase Authentication untuk login pengguna, termasuk integrasi akun Google, serta Firebase Realtime Database untuk penyimpanan data, dengan penambahan library dan plugin pada build.gradle.kts (module dan project).

```
implementation(platform(libs.firebase.bom))
implementation(libs.firebase.analytics)
implementation(libs.firebase.auth)
implementation(libs.androidx.credentials)
implementation(libs.androidx.credentials.play.services.auth)
implementation(libs.googleid)
implementation(libs.firebase.database)
implementation(libs.google.firebase.auth)
implementation(libs.play.services.auth)
```

Gambar 9 Implementasi Library Firebase

```
plugins {
    alias(libs.plugins.android.application)
    alias(libs.plugins.kotlin.android)
    id("com.google.gms.google-services")
    alias(libs.plugins.compose.compiler)
```

Gambar 10 Implementasi Plugins Firebase pada Module Level

```
plugins {
    alias(libs.plugins.android.application) apply false
    alias(libs.plugins.kotlin.android) apply false
    id("com.google.gms.google-services") version "4.4.3" apply false
}
```

Gambar 11 Implementasi Plugins Firebase Pada Project Level

3) Implementation Dependency Injection

```
package com.example.ampiproject.di.modules

import ...

// Modul Koin untuk dependensi data dan domain
val dataModule = module {
    single { FirebaseAuth.getInstance() }
    single { FirebaseDatabase.getInstance() }
    single<AuthRepository> { AuthRepositoryImpl(get(), get()) }
    single<UserRepository> { UserRepositoryImpl(get()) }
    single<FinancialRepository> { FinancialRepositoryImpl(get(), get()) }
    single<TimeManagementRepository> { TimeManagementRepositoryImpl(get()) }
}

val domainModule = module {
    factory { LoginUseCase(get()) }
    factory { RegisterUseCase(get()) }
    factory { SignInWithGoogleUseCase(get()) }
    factory { SaveUserUseCase(get()) }
    factory { GetCurrentUserUseCase(get()) }
    factory { CreateTransactionUseCase(get()) }
    factory { GetTransactionsUseCase(get()) }
    factory { CreateOrUpdateBudgetUseCase(get()) }
    factory { GetBudgetUseCase(get()) }
    factory { UpdateBudgetUsageUseCase(get()) }
    factory { GetMonthlyFinancialSummaryUseCase(get()) }
    factory { GetMonthlySpendingChartDataUseCase(get()) }
    factory { GetDailySpendingSummaryUseCase(get()) }
```

Gambar 12 Implementasi Depedency Injection Koin Module

AuthModule Koin digunakan untuk mendeklarasikan dependency injeksi tanpa membuat instance manual, terbagi menjadi tiga: dataModule menyediakan Firebase Authentication dan Realtime Database, domainModule menyediakan use case dengan instance baru saat dibutuhkan, dan presentationModule menyediakan ViewModel untuk di-inject ke UI.

4) Implementasi Fitur Login dan Registrasi

Fitur login dan registrasi digunakan agar pengguna dapat mengakses aplikasi secara aman menggunakan akun pribadi seperti akun Google. Implementasi ini memanfaatkan Firebase Authentication untuk otentikasi dan Firebase Realtime Database untuk menyimpan data pengguna.

```
override suspend fun loginWithEmail(email: String, password: String): Result<User> = try {
    val authResult = auth.signInWithEmailAndPassword(email, password).await()
    val firebaseUser = authResult.user
    if (firebaseUser != null) {
        val userRef = database.getReference( path: "users").child(firebaseUser.uid)
        val snapshot = userRef.get().await()
        val username = snapshot.child( path: "username").getValue(String::class.java)
        val user = User(firebaseUser.uid, email username ?: "", username firebaseUser.email ?: "")
        Result.Success(user)
    } else {
        Result.Error(Exception("Login gagal"))
    }
} catch (e: Exception) {
    println("LOGIN ERROR DETAIL: ${e.message}") // Debug
    val message = when {
        e.message?.contains( other: "badly formatted", ignoreCase = true) == true -> "Format email tidak valid"
        e.message?.contains( other: "no user record", ignoreCase = true) == true -> "Email tidak terdaftar"
        e.message?.contains( other: "password is invalid", ignoreCase = true) == true -> "Password salah"
        e.message?.contains( other: "auth credential is incorrect", ignoreCase = true) == true -> "Email atau Password salah"
        else -> "Terjadi kesalahan, coba lagi"
    }
    Result.Error(Exception(message))
}
```

Gambar 13 Implementasi Data Layer Login Repository

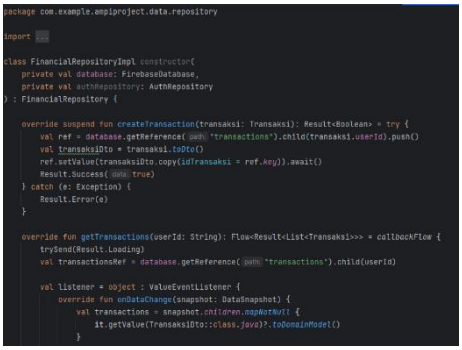
```
override suspend fun registerWithEmail(email: String, password: String): Result<User> = try {
    val authResult = auth.createUserWithEmailAndPassword(email, password).await()
    val firebaseUser = authResult.user
    if (firebaseUser != null) {
        val user = User(firebaseUser.uid, email "", username firebaseUser.email ?: "")
        Result.Success(user)
    } else {
        Result.Error(Exception("Registration failed"))
    }
} catch (e: Exception) {
    Result.Error(e)
}

override suspend fun signInWithGoogle(account: GoogleSignInAccount): Result<User> = try {
    val credential = GoogleAuthProvider.getCredential(account.idToken, null)
    val authResult = auth.signInWithCredential(credential).await()
    val firebaseUser = authResult.user
    if (firebaseUser != null) {
        val user = User(firebaseUser.uid, email firebaseUser.displayName ?: "", username firebaseUser.email ?: "")
        Result.Success(user)
    } else {
        Result.Error(Exception("Google Sign-In failed"))
    }
} catch (e: Exception) {
    Result.Error(e)
}
```

Gambar 14 Implementasi Data Layer Registrasi Repository

5) Implementasi Halaman Manajemen Keuangan

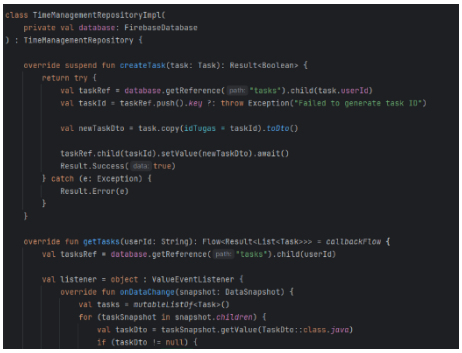
Halaman manajemen keuangan merupakan sebuah halaman yang menampilkan data keuangan harian dan bulanan pengguna, halaman ini menyediakan sebuah chart yang berguna untuk melihat sebuah chart data terkait manajemen keuangan yang dihitung berdasarkan jumlah nominal transaksi pengguna dari kategori pengeluaran pengguna. Berikut merupakan potongan kode-kode dalam implementasi fitur manajemen keuangan:



Gambar 15 Implementasi Data Layer Repository Manajemen Keuangan

6) Implementasi Halaman Manajemen Waktu

Halaman manajemen waktu merupakan sebuah halaman yang menampilkan data tugas harian dan bulanan pengguna, halaman ini menyediakan sebuah kalender yang berguna untuk melihat sebuah data kegiatan terkait manajemen waktu yang diklasifikasikan dalam kuadran manajemen waktu. Berikut merupakan potongan kode-kode dalam implementasi fitur manajemen keuangan:



Gambar 16 Implementasi Data Layer Repository Manajemen Waktu

4.2. Pengujian

selanjutnya ada pengujian yang dilakukan dengan menggunakan pengujian blackbox. Berikut beberapa pengujian yang diuji dalam implementasi aplikasi.

1) Pengujian Halaman Login

Tabel 1 Pengujian Halaman Login

Skenario Uji	Input	Hasil Yang Diharapkan	Hasil Pengujian	Status
Email & password benar	Email : user@mail.com Password: 12345678	Berhasil masuk ke halaman utama	Sesuai	Valid
Kesalahan email atau password	Email : salah@mail.com Password: salahpass	Menampilkan pesan “Email atau Password salah”	Sesuai	Valid
Kolom Kosong	Email: - Password: -	Menampilkan pesan kesalahan pada masing-masing field	Sesuai	Valid
Login dengan akun google	Memilih akun google	Berhasil masuk ke halaman utama	Sesuai	Valid

2) Pengujian Halaman Registrasi

Tabel 2 Pengujian Halaman Registrasi

Skenario Uji	Input	Hasil Yang Diharapkan	Hasil Pengujian	Status
Data Valid	Nama: User Email : user@mail.com Password: 12345678	Berhasil masuk ke halaman utama	Sesuai	Valid
Email Sudah digunakan	Nama: User Email : user@mail.com Password: 12345678	Menampilkan pesan “Email sudah digunakan”	Sesuai	Valid
Kolom Kosong	Nama: - Email: - Password: -	Menampilkan pesan kesalahan pada masing-masing field	Sesuai	Valid

3) Pengujian Fitur Manajemen Keuangan

Tabel 3 Pengujian Fitur Manajemen Keuangan

Skenario Uji	Input	Hasil Yang Diharapkan	Hasil Pengujian	Status
Data Valid	Tipe Transaksi: Pengeluaran Kategori: Makanan Nominal: 15.000 Deskripsi: Beli makan siang.	Data berhasil masuk ke database dan ditampilkan di halaman manajemen keuangan	Sesuai	Valid
Edit Data	Tipe Transaksi: Pengeluaran Kategori: Makanan Nominal: 15.000 Deskripsi: beli makan pagi	Data berhasil diubah dan disimpan ke dalam database serta di tampilkan di halaman manajemen keuangan	Sesuai	Valid
Kolom Kosong	Kategori: - Nominal: - Deskripsi: -	Menonaktifkan tombol simpan transaksi	Sesuai	Valid
Hapus Data	Klik timbol hapus	Menampilkan pesan pemberitahuan apakah ingin menghapus data atau tidak	sesuai	valid

4) Pengujian Fitur Manajemen Waktu

Tabel 4 Pengujian Fitur Manajemen Waktu

Skenario Uji	Input	Hasil Yang Diharapkan	Hasil Pengujian	Status
Data Valid	Nama Tugas: Belajar kotlin Deskripsi: Belajar kotlin jetpack compose Tanggal Tenggat: 15 Agustus 2023	Data berhasil masuk ke database dan ditampilkan di halaman manajemen waktu	Sesuai	Valid
Edit Data	Nama Tugas: Belajar kotlin Deskripsi: Belajar kotlin jetpack compose Tanggal Tenggat: 16 Agustus 2023	Data berhasil diubah dan disimpan ke dalam database serta di tampilkan di halaman manajemen keuangan	Sesuai	Valid
Kolom Kosong	Nama Tugas: - Deskripsi: - Tanggal Tenggat: -	Menonaktifkan tombol simpan tugas	Sesuai	Valid
Hapus Data	Klik tombol “Hapus”	Menampilkan pesan pemberitahuan apakah ingin menghapus data atau tidak	Sesuai	Valid

4.4 Pembahasan

Hasil pengujian melalui metode *black-box* mengonfirmasi bahwa seluruh fitur lanjutan pada Aplikasi Manajemen Personal Individual (AMPI) berfungsi sesuai dengan spesifikasi kebutuhan yang telah dirancang. Keberhasilan integrasi manajemen keuangan dan manajemen waktu dalam satu platform menunjukkan bahwa pendekatan holistik dapat menjawab

kompleksitas kebutuhan generasi muda dalam menjaga produktivitas dan kesehatan finansial secara simultan.

Analisis Arsitektur dan Keamanan Data Penerapan arsitektur *Clean MVVM* terbukti memberikan pemisahan yang jelas antara logika bisnis, data, dan antarmuka. Penggunaan *library* Koin sebagai *Dependency Injection* meminimalkan *boilerplate code*, yang secara signifikan meningkatkan modularitas dan kemudahan dalam pemeliharaan sistem (*maintainability*). Integrasi Firebase Authentication dan Realtime Database memastikan bahwa data pengguna tersinkronisasi secara instan lintas perangkat dengan standar keamanan yang andal. Hal ini menjawab kelemahan pada aplikasi AMPI versi awal yang masih mengandalkan penyimpanan lokal (*Room Database*), sehingga kini pengguna tidak perlu khawatir kehilangan data saat berganti perangkat.

Kontribusi Terhadap Penelitian Terdahulu Penelitian ini memberikan penguatan signifikan terhadap beberapa temuan terdahulu sekaligus menawarkan pembaruan fungsional.

1) Penguatan terhadap Literasi Keuangan: Temuan ini memperkuat penelitian Prameswari dkk. [7] dan Albadri dkk. [9] yang menekankan pentingnya pencatatan keuangan berbasis kategori untuk mengontrol pengeluaran. Namun, penelitian ini melangkah lebih jauh dengan menyatukan aspek keuangan tersebut dengan manajemen waktu, sebuah integrasi yang tidak dibahas secara mendalam pada penelitian-penelitian tersebut.
2) Validasi Arsitektur Modern: Hasil penelitian ini selaras dengan studi Ardiansyah dkk. [10] dan Prasetya & Aditya [11] yang menyatakan bahwa arsitektur *MVVM* dan *Clean Architecture* meningkatkan keteraturan kode pada aplikasi finansial. Implementasi dalam AMPI membuktikan bahwa struktur ini tidak hanya teoretis, tetapi secara praktis mampu menangani dua domain data yang berbeda (keuangan dan tugas) tanpa terjadi konflik data.
3) Efektivitas Manajemen Waktu: Penelitian ini mempertegas hasil studi Nugroho dkk. [14] mengenai efektivitas metode *waterfall* dalam membangun fitur pengingat tugas harian. Fitur notifikasi real-time dan klasifikasi prioritas tugas pada AMPI memberikan bukti empiris tambahan bahwa intervensi teknologi melalui *mobile reminder* sangat krusial dalam memitigasi kebiasaan menunda pekerjaan (*prokrastinasi*) pada remaja, sebagaimana diidentifikasi dalam analisis strategi manajemen waktu oleh Kifayatunnisa dkk. [2].
4) Inovasi Integrasi (Penyatuan Temuan): Berbeda dengan Pramudya & Edi [13] yang hanya fokus pada pelacakan kebiasaan (*habit tracking*), AMPI menyatukan visualisasi data keuangan (grafik donut) dengan manajemen jadwal. Hal ini menciptakan ekosistem manajemen personal yang lebih komprehensif. Integrasi ini memberikan kontribusi baru bagi bidang ilmu informatika terapan, khususnya dalam pengembangan sistem pendukung keputusan personal (*Personal Decision Support System*), di mana pengguna dapat melihat korelasi antara produktivitas waktu dan stabilitas ekonomi mereka dalam satu dasbor.

Secara keseluruhan, penelitian ini tidak hanya sekadar membangun aplikasi fungsional, tetapi juga mengonsolidasi berbagai temuan terpisah dari studi-studi sebelumnya menjadi sebuah solusi teknologi yang terpadu. Penguatan terhadap penggunaan teknologi *cloud-native* (Firebase) dan arsitektur bersih (*Clean Architecture*) memposisikan AMPI sebagai model pengembangan aplikasi manajemen personal yang relevan dengan kebutuhan generasi digital saat ini.

5. Simpulan

Berdasarkan hasil penelitian dan pengujian yang telah dilakukan, dapat disimpulkan bahwa Aplikasi Manajemen Personal Individual (AMPI) berbasis Android telah berfungsi dengan baik sesuai rancangan. Seluruh fitur utama, meliputi manajemen akun, manajemen keuangan, dan manajemen waktu, berjalan optimal dengan tingkat keberhasilan 100% melalui pengujian black-box tanpa ditemukan error fungsional.

Penerapan arsitektur Clean MVVM serta integrasi Firebase Authentication dan Firebase Realtime Database mampu menghasilkan sistem yang stabil, responsif, dan mudah dikembangkan. Aplikasi ini efektif membantu pengguna dalam mencatat pemasukan dan pengeluaran, memvisualisasikan kondisi keuangan melalui grafik, mengelola jadwal aktivitas, serta menerima pengingat tugas secara *real time*.

Dengan demikian, tujuan penelitian dan pengembangan aplikasi telah tercapai, di mana AMPI dapat memberikan manfaat nyata dalam mendukung pengelolaan aktivitas dan keuangan pribadi, khususnya bagi pengguna muda yang membutuhkan solusi terintegrasi dalam manajemen waktu dan keuangan sehari-hari.

Daftar Referensi

[1] S. A. Albadry, W. Pratiwi, S. Rusnaini, N. Istianingsih, N. T. Gonjales, and S. Alfiyah, "Pengaruh Perilaku Keuangan dan Financial Management terhadap Financial Sustainability pada Generasi Z dan Millenial di Kabupaten Bungo," *Jurnal EMT KITA*, vol. 9, no. 1, pp. 46–54, Dec. 2024.

[2] R. A. Kifayatunnisa, I. Saripah, and N. A. Nadhirah, "Analisis Strategi Manajemen Waktu pada Remaja," *Jurnal Bimbingan dan Konseling Pandohop*, 2024.

[3] Santi Puspa Oktaviane and Patah Herwanto, "Dampak Penggunaan Perangkat Mobile dalam Mendukung Kegiatan Pembelajaran Mandiri Siswa Kelas IX di SMP PGRI Rancaekek," *Uranus: Jurnal Ilmiah Teknik Elektro, Sains dan Informatika*, vol. 2, no. 4, pp. 165–174, Dec. 2024.

[4] Universitas Hayam Wuruk Perbanas, *Modul Lengkap - Pengantar Sistem Informasi (1)*. Universitas Hayam Wuruk Perbanas, 2022.

[5] V. Putuhuhu, "Sistem Informasi Manajemen Penelitian dan Pengabdian PNPB pada Politeknik Negeri Ambon," *Jurnal Simetrik*, vol. 12, no. 1, Jun. 2022.

[6] H. Sulaeman and A. Fira Waluyo, "KLIK: Kajian Ilmiah Informatika dan Komputer Perancangan Aplikasi Manajemen Keuangan Berbasis Mobile Menggunakan React Native Untuk Meningkatkan Literasi Keuangan Individu," *KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 4, no. 2, pp. 1021–1031, 2023.

[7] P. Prameswari, R. Mai Candra, M. Affandes, and L. Oktavia, "Desain Ui/Ux Aplikasi Manajemen Keuangan Pribadi Menggunakan Metode User Centered Design (UCD)," *Jurnal Pendidikan dan Teknologi Indonesia*, vol. 5, no. 1, Jan. 2025.

[8] N. S. Putri and F. R. Fadillah, "Integrasi Firebase Authentication dan Realtime Database pada Aplikasi Mobile Berbasis Android," *Jurnal Teknologi Informasi dan Komputer (JTIK)*, vol. 5, no. 3, pp. 211–220, 2024.

[9] A. F. Albadri, F. R. Firdaus, and K. Akbar, "Pengembangan Sistem Saran Keuangan Untuk Mahasiswa (Anak Kos) Berbasis Mobile Android," *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 3, no. 1, pp. 1–7, Oct. 2023.

[10] M. D. Ardiansyah, R. A. Kurniawan, and L. Prasetyo, "Implementasi Arsitektur MVVM pada Aplikasi Manajemen Keuangan Pribadi Berbasis Android," *Jurnal Teknologi dan Sistem Informasi*, vol. 13, no. 1, pp. 55–64, Mar. 2024.

[11] D. H. Prasetya and I. K. Aditya, "Analisis dan Implementasi Clean Architecture pada Aplikasi Mobile Keuangan Pribadi," *Jurnal Rekayasa dan Teknologi Informasi*, vol. 8, no. 1, pp. 87–98, Jan. 2025.

[12] H. A. Milkham, P. Sokibi, and A. Amroni, "Rancang Bangun Sistem Informasi Manajemen Keuangan Pribadi Menggunakan Metode Budget Jar Berbasis Android," *JSTIE (Jurnal Sarjana Teknik Informatika) (E-Journal)*, vol. 12, no. 1, p. 10, Feb. 2024.

[13] K. Pramudya and D. Edi, "Pengembangan Aplikasi Mobile HabiTroops," 2023.

[14] R. P. Nugroho, A. R. Wicaksono, and N. P. Lestari, "Pengembangan Aplikasi Manajemen Waktu dan Produktivitas Menggunakan Metode Waterfall," *Jurnal Ilmiah Informatika dan Rekayasa Perangkat Lunak*, vol. 4, no. 2, pp. 145–154, Jun. 2023.

[15] J. J. Sanjaya and J. Susilo, "Perbandingan Performa Kotlin vs Java dalam Pengembangan Android dengan Metode Iterasi While," *bit-Tech*, vol. 7, no. 2, pp. 545–553, Dec. 2024.

[16] Y. Anis, A. B. Mukti, and A. N. Rosyid, "Penerapan Model Waterfall Dalam Pengembangan Sistem Informasi Aset Destinasi Wisata Berbasis Website," *KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 4, no. 2, pp. 1134–1142, Oct. 2023.