Programming Certification Report

Oleh:

Rae Kenneth - 0706022110018

UNIVERSITAS CIPUTRA

SURABAYA

Daftar Isi

1. Pendahuluan

Aplikasi ini adalah Sistem Manajemen Perpustakaan yang dirancang untuk mempermudah pengelolaan buku, anggota, dan peminjaman di perpustakaan. Aplikasi ini dibangun menggunakan Swift dan SwiftUI untuk pengembangan antarmuka, SQLite sebagai database, dan menerapkan pola arsitektur MVVM (Model-View-ViewModel). Sistem ini mendukung operasi CRUD (Create, Read, Update, Delete) dan relasi antara berbagai entitas utama seperti buku, anggota, dan peminjaman.

2. Perancangan Sistem

2.1. Arsitektur Aplikasi (MVVM)

Aplikasi ini menggunakan pola arsitektur MVVM, yang bertujuan untuk memisahkan logika bisnis dari antarmuka pengguna:

- **Model:** Berfungsi untuk mengelola data aplikasi serta menangani komunikasi dengan database SQLite.
- **ViewModel:** Bertanggung jawab untuk memformat data dan mengelola logika aplikasi, termasuk operasi CRUD.
- **View:** Berperan dalam menampilkan data kepada pengguna sekaligus menangkap interaksi yang dilakukan pengguna.

2.2. Diagram Entitas dan Relasi (ERD)

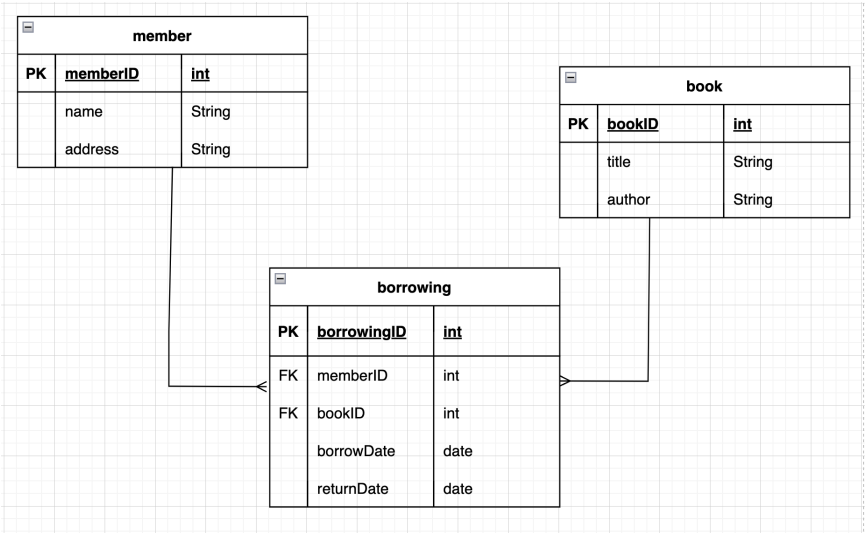


Diagram Entitas dan Relasi (ERD) menggambarkan relasi antara tabel berikut:

- **Book:** Menyimpan informasi buku.
- **Member:** Menyimpan data anggota perpustakaan.
- **Borrowings:** Melacak peminjaman buku oleh anggota.

2.3. Diagram Kelas

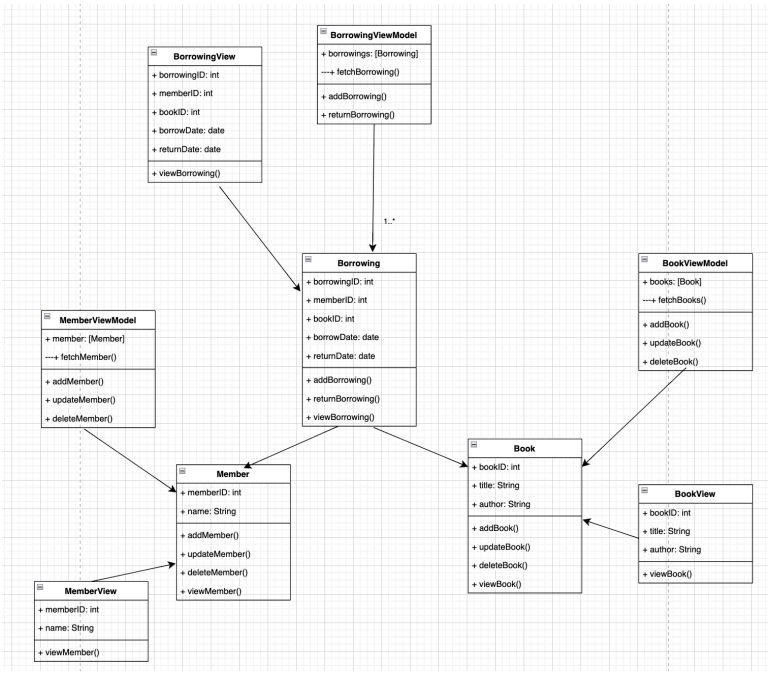
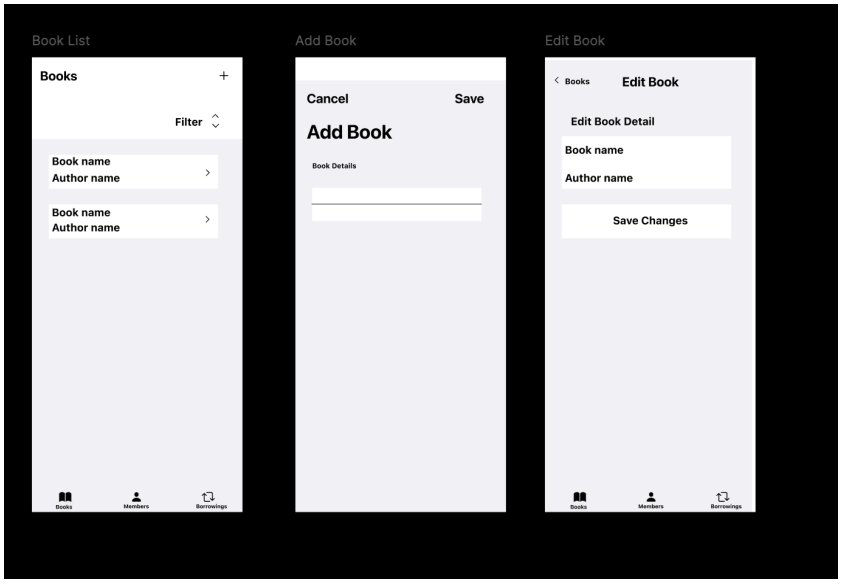


Diagram kelas menunjukkan hubungan antar komponen utama, seperti BookViewModel, MemberViewModel, dan BorrowingViewModel, serta asosiasi antar entitas seperti Book, Member, dan Borrowing.

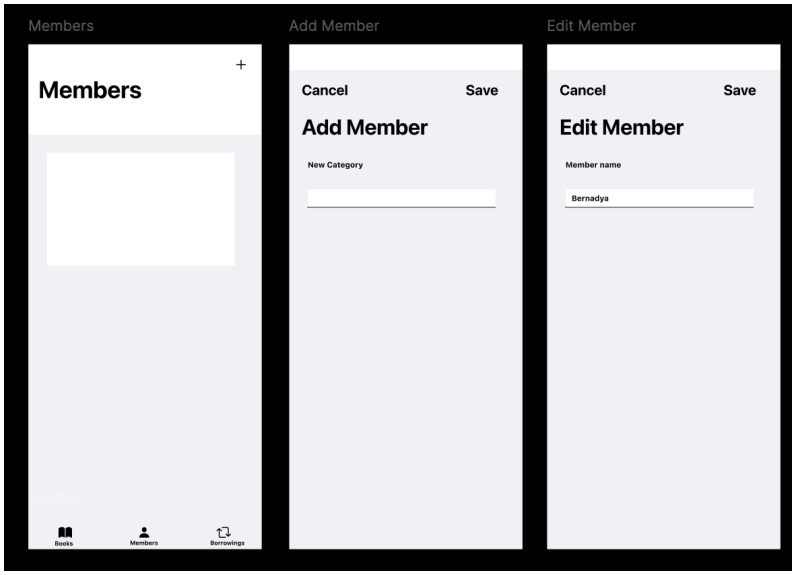
2.4. Wireframe Aplikasi

Berikut merupakan wireframe yang mencakup antarmuka pengguna untuk fitur utama:

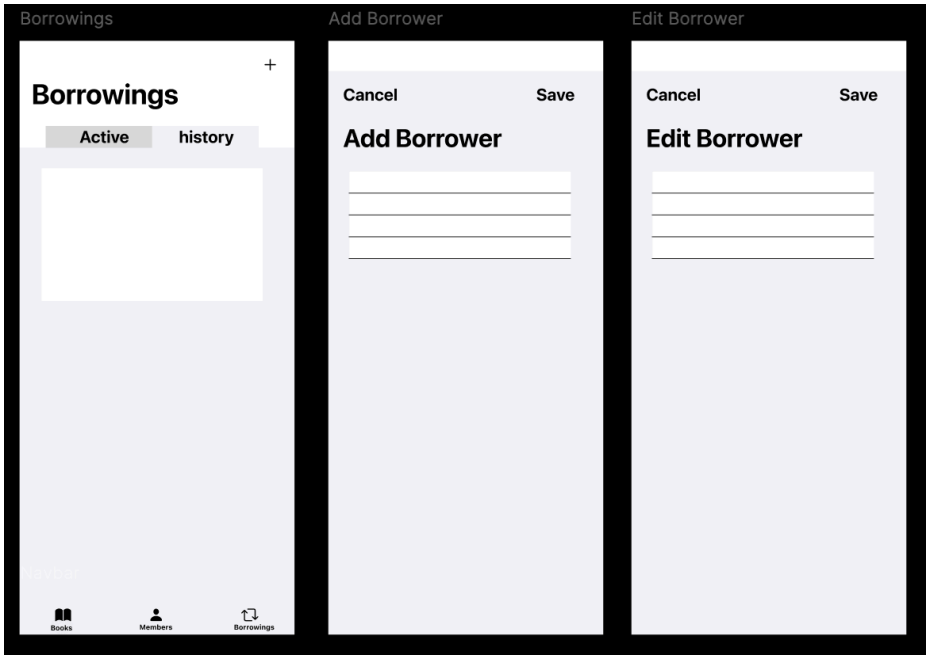
- **CRUD Buku**



- **CRUD Anggota**



- **Manajemen Peminjaman Buku**



3. Implementasi

3.1. Struktur Database SQLite

Tabel yang digunakan dalam SQLite adalah Books, Members, Borrowings, dan Relasi antara tabel mengikuti desain ERD.

3.2. Langkah Implementasi

1. **Pembuatan Proyek** Membuat proyek baru di XCode dengan framework SwiftUI.
2. **Koneksi ke SQLite** Membuat kelas SQLite Manager untuk menginisialisasi database, membuat tabel, dan menjalankan operasi CRUD (seperti INSERT, UPDATE, DELETE, dan SELECT).
3. **Membuat Model** Model mencerminkan tabel dalam database, seperti Book, Member, dan Borrowing, **Membuat ViewModel** ViewModel menghubungkan model dengan tampilan dan menangani logika bisnis.
4. **Membuat View** Antarmuka pengguna dibuat menggunakan SwiftUI, dengan fokus pada kemudahan penggunaan dan responsivitas.

4. Pengujian

4.1. Metode Pengujian

Metode pengujian menggunakan Black Box Testing dengan skenario seperti pengelolaan buku, anggota, dan peminjaman.

No	Fitur	Skenario	Hasil Diharapkan	Hasil Pengujian
1	Manajemen Buku	Tambah, edit, hapus, dan lihat buku.	Buku berhasil dikelola.	Buku berhasil dikelola.
2	Manajemen Anggota	Tambah, edit, hapus, dan lihat anggota.	Anggota berhasil dikelola.	Anggota berhasil dikelola.
3	Relasi Peminjaman	Tampilkan buku yang dipinjam oleh anggota tertentu.	Buku yang dipinjam berhasil tampil.	Buku yang dipinjam berhasil tampil.
4	Validasi Input	Cegah data kosong saat input buku, atau anggota.	Pesan kesalahan ditampilkan.	Pesan kesalahan ditampilkan.

5. Debugging dan Dokumentasi

5.1. Langkah Debugging untuk Penanganan Kesalahan

1. **Identifikasi Kesalahan:**
    - Mengguakan breakpoints pada kode untuk memeriksa nilai variabel dan alur eksekusi.
2. **Penanganan Kesalahan Logika:**
    - Uji unit pada setiap metode ViewModel, seperti validasi input atau pengelolaan relasi.
3. **Penanganan Galat (Error Handling):**
    - Menggunakan blok do-try-catch untuk menangkap kesalahan saat berinteraksi dengan database SQLite.

```
private func openDatabase() {
    do {
        let fileURL = try FileManager.default
            .url(for: .documentDirectory, in: .userDomainMask, appropriateFor: nil, create: false)
            .appendingPathComponent("ManagementLibrary.sqlite")
        print("Database path: \(fileURL.path)")

        if sqlite3_open(fileURL.path, &db) != SQLITE_OK {
            print("Error opening database: \(String(cString: sqlite3_errmsg(db)))")
        } else {
            print("Database opened successfully")
        }
    } catch {
        print("Error locating database file: \(error.localizedDescription)")
    }
}
```

6. Kesimpulan

Proyek aplikasi Manajemen Perpustakaan Sederhana telah sukses dikembangkan menggunakan teknologi seperti Swift UI, SQLite, serta pola arsitektur MVVM. Pengelolaan koneksi ke SQLite dilakukan secara efisien melalui Database Manager, mencakup proses pembuatan tabel yang dirancang sesuai dengan ERD. Model aplikasi menggambarkan entitas utama seperti Buku, Anggota, dan peminjaman yang diintegrasikan melalui relasi One-to-Many dan Many-to-Many. Sebagai penghubung antara Model dan View, ViewModel menangani logika bisnis dan mendukung operasi CRUD secara efektif. Sementara itu, SwiftUI memungkinkan pembuatan antarmuka pengguna yang modern dan responsif, memanfaatkan mekanisme binding untuk memastikan tampilan selalu sinkron dengan perubahan data. Hasil pengujian menunjukkan aplikasi ini berjalan sesuai spesifikasi, termasuk validasi input yang telah diterapkan dengan baik. Proyek ini membuktikan bahwa pendekatan pengembangan yang terstruktur, didukung oleh teknologi yang tepat, dapat menghasilkan aplikasi yang fungsional, andal, dan sesuai dengan kebutuhan pengguna.