Laporan Akhir
Sertifikasi

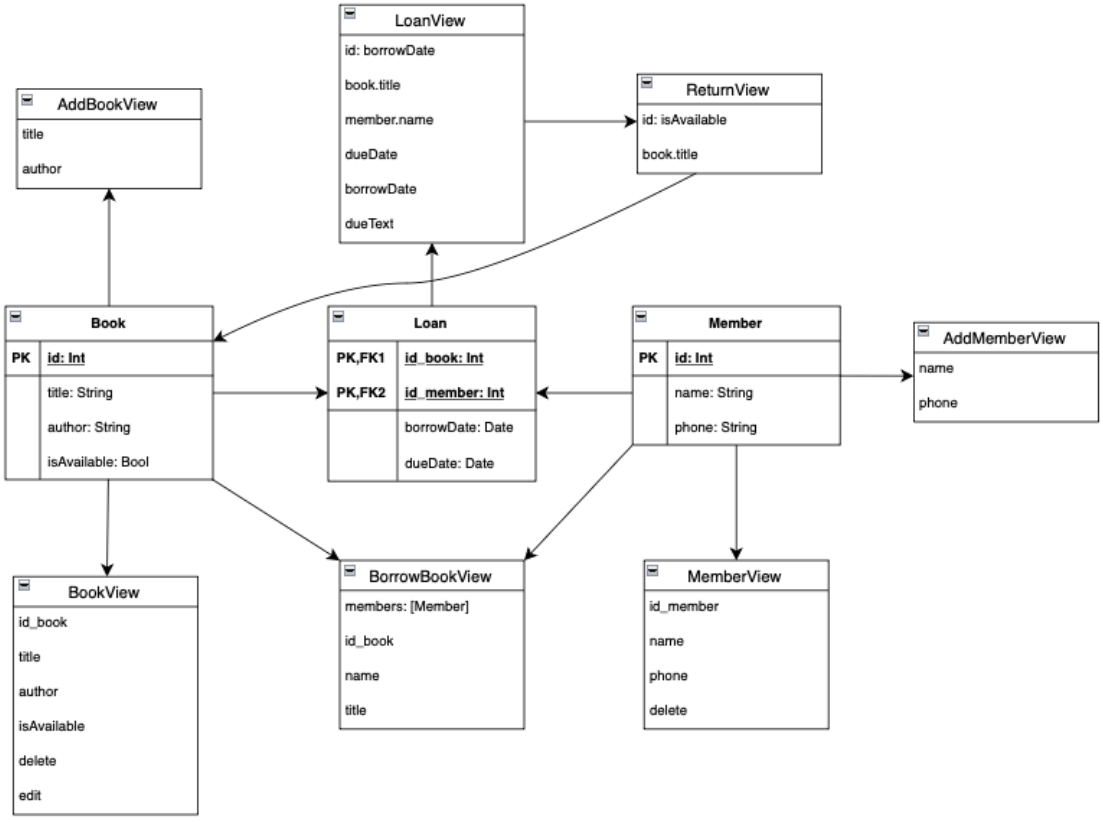
Trisha Alexis Likorawung
0706022110037

Information Systems for Business
Universitas Ciputra Surabaya
2024/2025

Daftar Isi

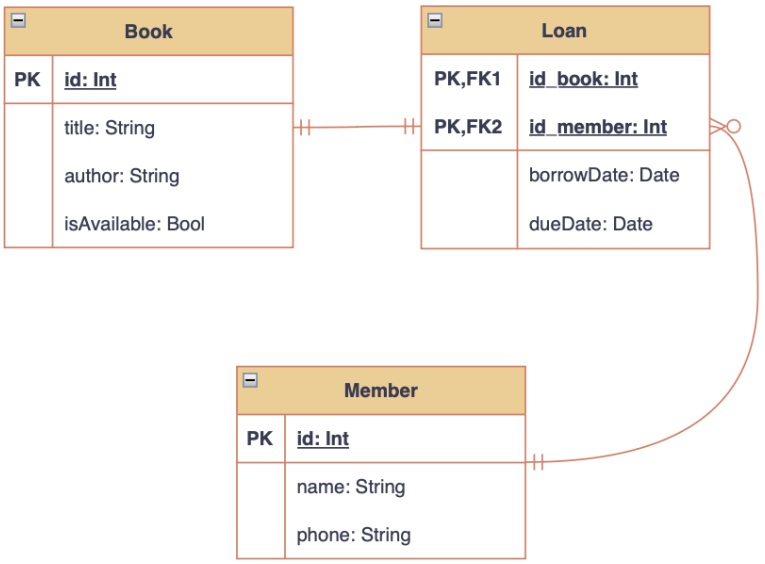
I. Design System

A. Class Diagram



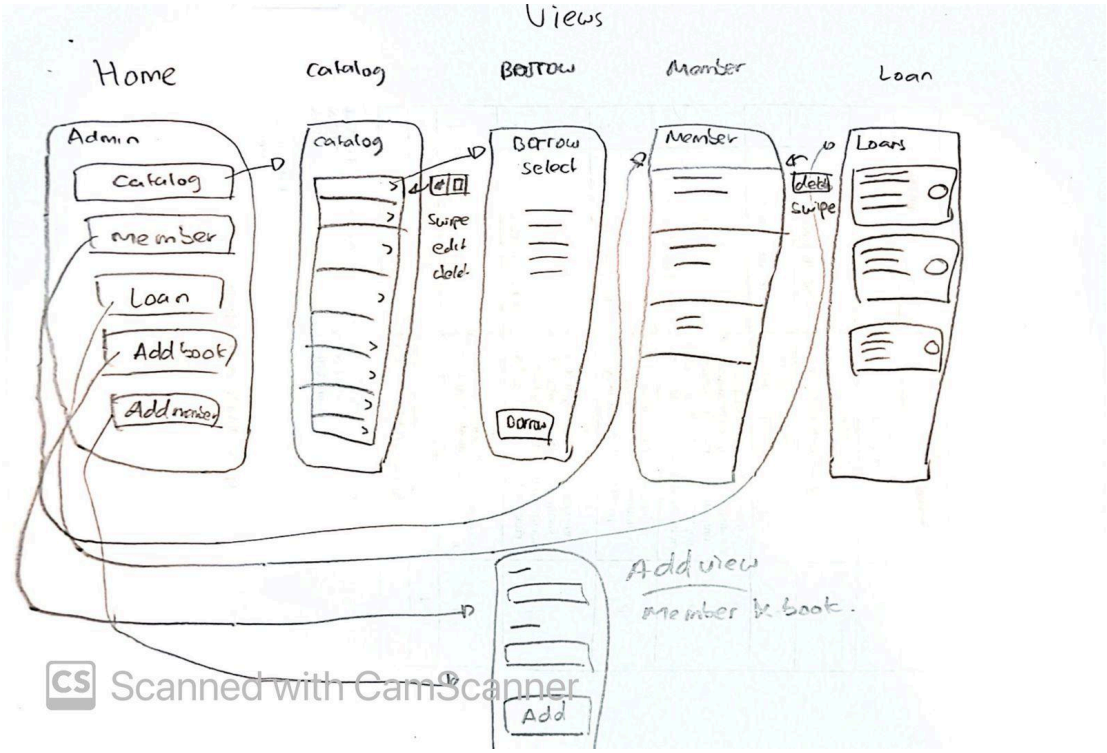
Gambar diatas merupakan hasil *Class Diagram* untuk objek yang ada pada aplikasi. *Class Diagram* di atas adalah hasil pembuatan *Class Diagram* untuk objek-objek yang tersedia dalam aplikasi. Diagram kelas ini menggambarkan struktur data dari aplikasi perpustakaan admin yakni aplikasi iOS. Saya menggunakan Swift Data. Kelas utama meliputi *Book, Member, dan Loan*. Selain itu, ada *View* yang mewakili data yang relevan dalam antarmuka pengguna seperti *BookView, LoanView dan MemberView*. Ada pula antarmuka lain seperti *Borrow Book View, Return View, Add Member View, dan Add Book View.* Kelas-kelas ini berisi atribut yang sesuai dengan data yang mereka wakili. Saya tidak membuat login untuk memudahkan pemeriksaan.

B. ERD



Gambar diatas merupakan ERD diagram aplikasi dimana table *book* merupakan *one to one* ke tabel *loan* karena hanya ada satu buku per judul kalau buku dipinjamkan jadi tidak *available* lagi. Lalu hubungan dengan *member* dan *loan one to many* karena satu member bisa pinjam lebih dari satu buku (bukan buku yang sama karena adanya hanya satu buku per judulnya). Buku dibuat tabel sederhana tapi bisa ditambahkan isbn, tahun terbit, penerbit, gambar, dan lain-lain bila ingin lebih lengkap. Begitu pula dengan member bisa ditambahkan tanggal lahir, asal, alamat, dan sebagainya kalau mau lebih lengkap datanya.

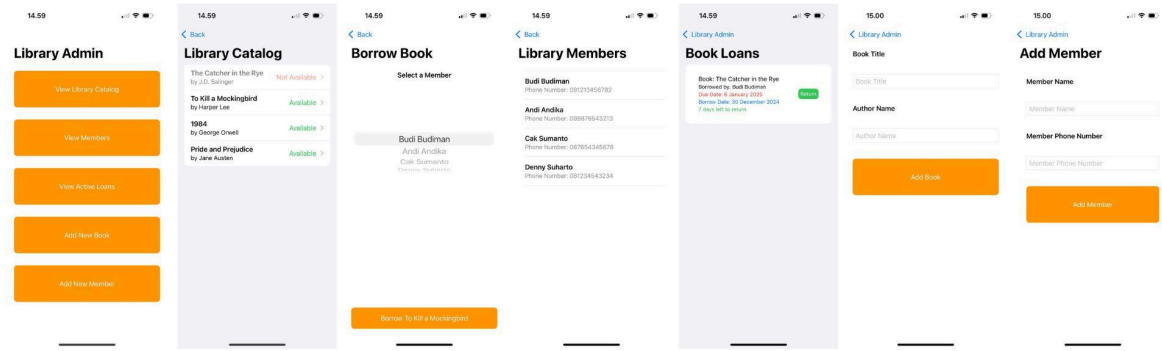
C. WireFrame



Gambar diatas merupakan hasil wireframing yang dibuat di kerta untuk merancang sketsa awal aplikasi berupa gambaran besar dari aplikasi yang akan dibuat. Terdapat halaman *home/admin dashboard* untuk menampilkan tombol-tombol yang dapat dilakukan. Terdapat katalog bisa lihat *list* judul buku, penulis dan *status* bisa dipinjam atau tidak. Kalau tidak maka akan tidak bisa dipinjam atau *not available.* Bisa edit buku, bisa hapus buku dengan di *swipe.* Kalau di tekan bisa ke *borrow book view* yang bisa pilih member mana yang akan pinjam dan tombol dipinjam maka akan berubah status dan *available* jadi *not available.*

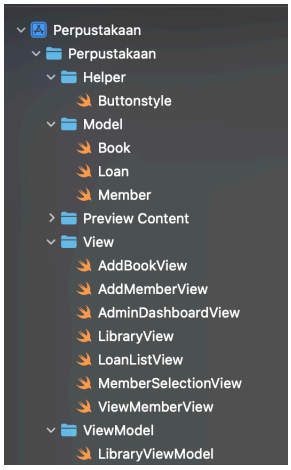
Bisa lihat member isinya *list* nama dan nomor telepon bisa *delete member*. Ada *view loans* bisa lihat active *loans* isinya judul buku, dipinjam oleh nama member, tanggal harus dikembalikan, tanggal dipinjam, berapa hari lagi di kembalikan, ada tombol *return* jadi kalau sudah kembali jadi available lagi *statusnya* lalu akan hilang dari *active loans.* Lalu, ada juga *add book dan add member* jadi bisa menambahkan member dan menambahkan buku.

Preview Aplikasi



II. Langkah Implementasi

1. Membuat *wireframe*
   Pertama-tama sebelum aplikasi dibuat saya membuatkan wireframing untuk merencanakan apa saja yang dibutuhkan di selembar kertas.
2. Membuat *design ERD*
   Setelah itu, saya membuat desain ERD untuk *database* aplikasi, yang kemudian diimplementasikan sebagai *swift data.*
3. Membuat *project* di *xcode* dan membuat *git* yang *public*
   Saya lalu membuat projectnya di xcode menggunakan *swiftui* untuk tampilan, *swift data* untuk lokal database, dan bisa *add, edit dan delete pakai function-function*.
4. Membuat MVVM
   Saya merapikan arsitektur dengan MVVM (model-view-viewmodel). Memisahkan jadi folder-folder, ada view yang isinya tampilan saja namanya diakhirinya dengan view seperti memberview, libraryview, dan lain-lain. Folder view model isinya yang ada function di masukkan disana. Kemudian databasenya di folder model ada *book, loan, dan member.* Ada pula helper yang isinya gaya tombol atau *button style*.

5. Membuat *error handling dan clean code*
   Saya membuat clean code dan error handling untuk tahu bug dimana dan pakai print, else, dan sebagainya.
6. Membuat Class Diagram

Selanjutnya saya membuat *class diagram* yakni hubungan antar objek pada aplikasi

7. Pengujian UI Testing
   Terakhir, saya melakukan unit testing dengan ui testing untuk tahu ada error apa dan hasilnya berjalan lancar saya menggunakan satu test case yakni menambahkan member baru. Selain itu, testing dilakukan secara manual dan berjalan lancar.

III. Hasil Pengujian

Test Case	Steps	Ekspektasi	Hasil
User membuat member baru	1. Pilih tombol add member 2. Ketikkan nama “Trisha” di textfield name 3. Ketikkan nomor telepon “081325634789” 4. Tekan add member	Akan melakukan semua perintah dan terakhir tombol add member ditekan	Pass (aplikasi juga melakukan serangkaian percobaan seperti dengan orientasi landscape, darkmode, dsb)

```swift
import XCTest

final class PerpustakaanUITests: XCTestCase {

    override class var runsForEachTargetApplicationUIConfiguration: Bool {
        true
    }

    override func setUpWithError() throws {
        continueAfterFailure = false
    }

    @MainActor
    func testAddMemberForm() throws {
        let app = XCUIApplication()
        app.launch()

        let launchScreenAttachment = XCTAttachment(screenshot: app.screenshot())
        launchScreenAttachment.name = "Launch Screen"
        launchScreenAttachment.lifetime = .keepAlways
        add(launchScreenAttachment)

        let addNewMemberButton = app.buttons["Add New Member"]
        XCTAssertTrue(addNewMemberButton.exists, "The 'Add New Member' button should exist")
        addNewMemberButton.tap()

        let addMemberTitle = app.staticTexts["Add Member"]
        XCTAssertTrue(addMemberTitle.exists, "The AddMemberView screen should be displayed after tapping 'Add New Member'")

        let nameTextField = app.textFields["Member Name"]
        XCTAssertTrue(nameTextField.exists, "The 'Member Name' text field should exist")
        nameTextField.tap()
        nameTextField.typeText("Trisha")

        let phoneTextField = app.textFields["Member Phone"]
        XCTAssertTrue(phoneTextField.exists, "The 'Member Phone' text field should exist")
        phoneTextField.tap()
        phoneTextField.typeText("081325634789")

        let addMemberButton = app.buttons["Add Member"]
        XCTAssertTrue(addMemberButton.exists, "The 'Add Member' button should exist")
        addMemberButton.tap()

        let successAlert = app.alerts["Member Added"]
        XCTAssertTrue(successAlert.exists, "The success alert should appear after adding a member")

        successAlert.buttons["OK"].tap()

        let addMemberFormScreenshot = XCTAttachment(screenshot: app.screenshot())
        addMemberFormScreenshot.name = "Add Member Form After Success"
        addMemberFormScreenshot.lifetime = .keepAlways
        add(addMemberFormScreenshot)
    }
}
```