Project Overview

Build an offline-capable iOS app with SwiftUI that allows users to:

- Register and login using secure credentials
- Add, view, and manage notes containing title, description, and photos
- Store all data locally on device with encrypted password storage
- Validate inputs strictly according to Indian standards and password policy
- Support all screen sizes and iOS versions (back to iOS 13+)
- Include unit testing (XC test case)

Architecture

- **MVVM (Model-View-ViewModel) Pattern**
  Separate concerns for UI, business logic, and data persistence
- **Data Persistence**
  Use **Core Data** for offline storage
  Passwords encrypted with **Keychain Services** or **CryptoKit**
- **UI Framework**
  SwiftUI for views and navigation
- **Image Storage**
  Store photos in app's documents directory, save URLs/paths in DB

Screens & Features

Screen	Inputs/Display	Actions
Login Screen	Mobile/Email, Password	Validate inputs, authenticate user
Signup Screen	Name, Mobile, Email, Password	Validate inputs, encrypt password, save user locally
Home Screen	List of Notes (Title, Description, Photo)	Tap row → navigate to Details screen
Details Screen	Show note's photos, title, description	-
Add Note Screen	Title, Description, Add photos (max 10)	Validate inputs, save note locally

Validations

Field	Validation Rules	Example Regex / Logic
Mobile	Indian number format, 10 digits, starts with 6-9	^[6-9]\d{9}$
Email	Regex-based email validation	^[A-Z0-9a-z._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$
Name	Min 4, max 25 characters, alphabets and spaces	^[A-Za-z]{4,25}$
Password	8-15 chars, first char lowercase, ≥2 uppercase, ≥2 digits, ≥1 special, no name substring	Custom logic with regex + string checks
Title	5-100 characters	String length check
Description	100-1000 characters	String length check
Photos	Max 10	Array count check

Offline Storage

- Use **Core Data** or **SQLite** for structured data storage
- Photos stored as files, paths saved in DB
- On app launch, load users and notes from local DB