Mindful

Guide Architectural de Référence

Kickstart du Projet - 3 Parties

Partie 2

Liquid Glass UI

Design System - Palette de Couleurs

```swift
extension Color {
    // Gradients principaux
    static let mindfulPrimary = Color(hex: "667eea")
    static let mindfulSecondary = Color(hex: "764ba2")

    // Glass effects
    static let glassBackground = Color.white.opacity(0.2)
    static let glassBorder = Color.white.opacity(0.6)

    // Moods
    static let moodVeryHappy = Color.green
    static let moodHappy = Color(hex: "90EE90")
    static let moodNeutral = Color.blue
    static let moodSad = Color.orange
    static let moodVerySad = Color.red
}
```

Design System - Typography

```swift
extension Font {
    // Headings
    static let entryTitle = Font.system(
        size: 28, weight: .bold, design: .rounded
    )
    static let cardTitle = Font.system(
        size: 18, weight: .semibold
    )

    // Body
    static let entryBody = Font.system(
        size: 16, weight: .regular
    )
    static let caption = Font.system(
        size: 14, weight: .regular
    )

    // Special
    static let date = Font.system(
        size: 12, weight: .medium, design: .monospaced
    )
}
```

Components à Créer

Core Components :

- `GlassCard` - Container de base
- `GlassButton` - Boutons avec effet glass
- `MoodSelector` - Picker d'humeur animé
- `TagPill` - Pills pour les tags
- `PhotoGallery` - Galerie de photos

Complex Components :

- `EntryCard` - Card pour timeline
- `EntryEditor` - Éditeur rich text
- `StatsCard` - Card pour statistiques
- `SearchBar` - Barre de recherche glass

GlassCard - Signature

```
struct GlassCard<Content: View>: View {
    let content: Content

    init(@ViewBuilder content: () -> Content)

    var body: some View {
        content
            .padding()
            .background(.ultraThinMaterial)
            .cornerRadius(20)
            .shadow(...)
            .overlay(borderGradient)
    }
}
```

Usage :

```
swift
GlassCard {
    Text("Hello")
}
```

MoodSelector - UI

```
[😄] [🙂] [😐] [😔] [😢]
  ↑
Selection avec animation scale + color highlight
```

Features :

- Sélection tactile
- Animation spring au tap
- Couleur de fond change selon mood
- Haptic feedback

Ressource :

- UIFeedbackGenerator pour haptics
- Human Interface Guidelines - Haptics

Views Architecture

```
MindfulApp
└── ContentView
    ├── TimelineView (liste des entrées)
    │   └── EntryCard (row)
    ├── EditorView (nouvelle/édition entrée)
    │   ├── MoodSelector
    │   ├── TextEditor
    │   └── PhotoGallery
    ├── DetailView (lecture entrée)
    │   └── EntryContent + Photos
    └── StatsView (dashboard)
        └── StatsCards
```

TimelineView - Structure

```
struct TimelineView: View {
    @State private var viewModel = TimelineViewModel()
    @Namespace private var animation

    var body: some View {
        ScrollView {
            LazyVStack(spacing: 20) {
                ForEach(viewModel.entries) { entry in
                    EntryCard(entry: entry)
                        .matchedGeometryEffect(
                            id: entry.id,
                            in: animation
                        )
                }
            }
        }
        .background(gradientBackground)
    }
}
```

EditorView - Features

Composants :
- `TextEditor` natif SwiftUI
- `MoodSelector` en header
- `PhotoPicker` (PhotosUI)
- `TagField` avec suggestions
- Toolbar glass en bas (save, cancel, add photo)

Auto-save :
- Debouncing avec Task cancellation
- Sauvegarde toutes les 2 secondes
- Indicateur "Saved" subtil

Writing Tools : Intégration automatique iOS 18+

Animations - Patterns

Hero Animation :
```swift
@Namespace private var animation


.matchedGeometryEffect(id: entry.id, in: animation)
```

Spring Animations :
```swift
withAnimation(.spring(response: 0.6, dampingFraction: 0.7)) {
    // state change
}
```

Phase Animator :
```swift
PhaseAnimator([false, true]) { phase in
    // animate through phases
}
```

Animations - Transitions

```
extension AnyTransition {
    static var slideAndFade: AnyTransition {
        .asymmetric(
            insertion: .move(edge: .trailing)
                       .combined(with: .opacity),
            removal: .move(edge: .leading)
                     .combined(with: .opacity)
        )
    }
}
```

Usage :

```
swift
if showDetail {
    DetailView()
        .transition(.slideAndFade)
}
```

Skeleton Loaders

```
struct SkeletonView: View {
    @State private var shimmerOffset: CGFloat = -1

    var body: some View {
        RoundedRectangle(cornerRadius: 12)
            .fill(.gray.opacity(0.3))
            .overlay(shimmerGradient)
            .onAppear {
                withAnimation(.linear(duration: 1.5)
                    .repeatForever(autoreverses: false)) {
                    shimmerOffset = 1
                }
            }
    }
}
```

Ressources UI - SwiftUI

Documentation officielle :
- SwiftUI Documentation
- SwiftUI Tutorials

WWDC Sessions :
- WWDC23: Animate with springs (10158)
- WWDC23: Demystify SwiftUI performance (10160)
- WWDC22: SwiftUI on iPad: Add toolbars (10069)

Communauté :
- Hacking with Swift - SwiftUI
- SwiftUI Lab

Ressources UI - Design

Apple Design Resources :

- Human Interface Guidelines
- SF Symbols App

Inspiration :

- Dribbble : recherche "journal app" ou "glass design"
- Mobbin : collection d'apps iOS design

Glassmorphism :

- Articles sur le style "Liquid Glass" / "Glassmorphism"
- Figma templates pour référence