

# React Native Income & Expense App - Complete Folder Structure

---

```
IncomeExpenseApp/
├── android/           # Android platform files
├── ios/               # iOS platform files
├── server/            # Development API server
│   ├── db.json        # json-server database
│   └── db.seed.json    # Seed data for reset
├── src/
│   ├── app/           # App initialization & providers
│   │   └── App.tsx     # Main app component -
│   └── navigation/    # Navigation configuration
│       ├── AppNavigator.tsx # Root navigator -
│       ├── TabNavigator.tsx # Bottom tab navigator -
│       ├── StackNavigator.tsx # Stack navigator -
│       ├── types.ts     # Navigation types
│       └── providers/    # Context providers
│           ├── ThemeProvider.tsx # Theme context -
│           └── StoreProvider.tsx # Redux provider -
│   ├── components/    # Reusable UI components
│   │   ├── ui/         # Basic UI components
│   │   │   ├── Button.tsx -
│   │   │   ├── Input.tsx -
│   │   │   ├── Card.tsx -
│   │   │   ├── Modal.tsx -
│   │   │   ├── ProgressBar.tsx -
│   │   │   ├── Chip.tsx -
│   │   │   ├── FloatingActionButton.tsx -
│   │   │   ├── LoadingSpinner.tsx -
│   │   │   └── EmptyState.tsx -
│   └── Done
└── Done
```

	└─ index.ts	# Exports	-
Done			
	└─ forms/	# Form components	
	└─ FormField.tsx		-
Done			
	└─ AmountKeypad.tsx		-
Done			
	└─ DatePicker.tsx		
	└─ CategorySelector.tsx		
	└─ AccountSelector.tsx		
	└─ TagInput.tsx		
	└─ charts/	# Chart components	
	└─ PieChart.tsx		
	└─ BarChart.tsx		
	└─ LineChart.tsx		
	└─ ChartContainer.tsx		
	└─ lists/	# List components	
	└─ TransactionItem.tsx		
	└─ AccountItem.tsx		
	└─ CategoryItem.tsx		
	└─ BudgetItem.tsx		
	└─ SectionHeader.tsx		
	└─ layout/	# Layout components	
	└─ SafeContainer.tsx		
	└─ Header.tsx		
	└─ TabBar.tsx		
	└─ KeyboardAvoidingContainer.tsx		
	└─ screens/	# Screen components	
	└─ auth/	# Authentication screens	
	└─ OnboardingScreen.tsx		
	└─ LockScreen.tsx		
	└─ dashboard/	# Dashboard screens	
	└─ DashboardScreen.tsx		
	└─ components/		
	└─ KPICards.tsx		
	└─ MonthSelector.tsx		
	└─ MiniCharts.tsx		
	└─ QuickActions.tsx		
	└─ transactions/	# Transaction screens	
	└─ TransactionListScreen.tsx		
	└─ TransactionDetailScreen.tsx		
	└─ AddTransactionScreen.tsx		
	└─ components/		
	└─ TransactionFilters.tsx		
	└─ TransactionForm.tsx		
	└─ TransactionTypeToggle.tsx		
	└─ AttachmentManager.tsx		
	└─ accounts/	# Account management	
	└─ AccountManagerScreen.tsx		
	└─ components/		
	└─ AccountForm.tsx		
	└─ AccountBalance.tsx		

```

├── categories/                                # Category management
│   ├── CategoryManagerScreen.tsx
│   └── components/
│       ├── CategoryForm.tsx
│       ├── CategoryTree.tsx
│       └── CategoryIcon.tsx
├── budgets/                                  # Budget screens
│   ├── BudgetScreen.tsx
│   └── components/
│       ├── BudgetForm.tsx
│       ├── BudgetProgress.tsx
│       └── BudgetAlerts.tsx
├── reports/                                 # Report screens
│   ├── ReportsScreen.tsx
│   └── components/
│       ├── ReportFilters.tsx
│       ├── MonthlyReport.tsx
│       ├── YearlyReport.tsx
│       └── ExportOptions.tsx
├── settings/                                # Settings screens
│   ├── SettingsScreen.tsx
│   └── components/
│       ├── SettingsItem.tsx
│       ├── CurrencySelector.tsx
│       ├── ThemeSelector.tsx
│       └── DataManagement.tsx
├── features/                                # Feature-based modules
│   ├── accounts/                            # Account feature
│   │   ├── hooks/
│   │   │   ├── useAccountBalance.ts
│   │   │   └── useAccountValidation.ts
│   │   ├── utils/
│   │   │   └── accountUtils.ts
│   │   └── types.ts
│   ├── categories/                          # Category feature
│   │   ├── hooks/
│   │   │   └── useCategoryHierarchy.ts
│   │   ├── utils/
│   │   │   └── categoryUtils.ts
│   │   └── types.ts
│   ├── transactions/                         # Transaction feature
│   │   ├── hooks/
│   │   │   ├── useTransactionForm.ts
│   │   │   ├── useTransactionFilters.ts
│   │   │   └── useTransactionValidation.ts
│   │   ├── utils/
│   │   │   ├── transactionUtils.ts
│   │   │   └── balanceCalculations.ts
│   │   └── types.ts
│   └── budgets/                              # Budget feature
│       ├── hooks/
│       │   └── useBudgetProgress.ts

```

```

├── useBudgetAlerts.ts
├── utils/
│   ├── budgetUtils.ts
│   ├── types.ts
├── reports/                                # Reports feature
│   ├── hooks/
│   │   ├── useReportData.ts
│   │   └── useReportFilters.ts
│   ├── utils/
│   │   ├── reportCalculations.ts
│   │   └── chartDataTransformers.ts
│   └── types.ts
├── state/                                # Redux state management
│   ├── store.ts                          # Store configuration
├── api.ts                                # RTK Query API slice
├── slices/                                # Redux slices
│   ├── appSlice.ts                       # App-wide state
│   ├── filtersSlice.ts                   # Filter state
│   ├── preferencesSlice.ts              # User preferences
│   └── uiSlice.ts                        # UI state
├── selectors/                             # Reselect selectors
│   ├── accountSelectors.ts
│   ├── transactionSelectors.ts
│   ├── budgetSelectors.ts
│   └── reportSelectors.ts
├── types.ts                              # Redux types
├── services/                             # External services
│   ├── api/                              # API services
│   │   ├── baseQuery.ts                 # Custom base query
│   │   ├── endpoints/                   # API endpoints
│   │   │   ├── accounts.ts
│   │   │   ├── categories.ts
│   │   │   ├── transactions.ts
│   │   │   └── budgets.ts
│   │   └── types.ts
│   ├── storage/                          # Storage services
│   │   ├── asyncStorage.ts
│   │   ├── fileStorage.ts
│   │   ├── encryption.ts
│   │   └── appInitialization.ts
├── localBaseQuery.ts

```

```

Done
├── notifications/                # Notification services
│   ├── localNotifications.ts
│   └── budgetAlerts.ts
├── export/                      # Export/Import services
│   ├── csvExport.ts
│   ├── csvImport.ts
│   └── backupService.ts
├── security/                   # Security services
│   ├── biometrics.ts
│   └── pinService.ts
├── attachments/               # File management
│   ├── imageService.ts
│   └── fileManager.ts
├── utils/                     # Utility functions
│   ├── constants/             # App constants
│   └── colors.ts
- Done
├── sizes.ts
├── fonts.ts
├── config.ts
├── helpers/                   # Helper functions
│   ├── dateUtils.ts
│   ├── currencyUtils.ts
│   ├── formatUtils.ts
│   ├── validationUtils.ts
│   └── deviceUtils.ts
├── hooks/                     # Custom hooks
│   ├── useDebounce.ts
│   ├── usePermissions.ts
│   ├── useKeyboard.ts
│   └── useAppState.ts
└── env.ts                     # Environment configuration
- Done
├── theme/                     # Theming
- Done
├── animations.ts              # Animation configs
- Done
├── colors.ts                  # Color palette
- Done
├── index.ts                   # Theme exports
- Done
├── shadows.ts                 # Shadow styles
- Done
├── spacing.ts                 # Spacing system
- Done
├── typography.ts              # Font styles
- Done
├── i18n/                      # Internationalization
└── index.ts                   # i18n configuration

```

└─ locales/	# Translation files
└─ en.json	
└─ bn.json	
└─ types.ts	# i18n types
└─ assets/	# Static assets
└─ images/	# Image assets
└─ icons/	# Custom icons
└─ illustrations/	# Illustrations
└─ splash/	# Splash screens
└─ fonts/	# Custom fonts
└─ data/	# Static data files
└─ categories.json	# Default categories
└─ currencies.json	# Currency data
└─ types/	# Global TypeScript types
└─ global.ts	# Global types
-Done	
└─ api.ts	# API response types
-Done	
└─ navigation.ts	# Navigation types
└─ theme.ts	# Theme types
└─ __tests__/	# Test files
└─ __mocks__/	# Test mocks
└─ components/	# Component tests
└─ screens/	# Screen tests
└─ utils/	# Utility tests
└─ features/	# Feature tests
└─ e2e/	# End-to-end tests
└─ docs/	# Documentation
└─ API.md	# API documentation
└─ COMPONENTS.md	# Component documentation
└─ TESTING.md	# Testing guide
└─ DEPLOYMENT.md	# Deployment guide
└─ .env	# Environment variables
└─ .env.example	# Environment template
└─ package.json	# Dependencies
└─ tsconfig.json	# TypeScript config
└─ metro.config.js	# Metro bundler config
└─ babel.config.js	# Babel config
└─ jest.config.js	# Jest config
└─ .eslintrc.js	# ESLint config
└─ .prettierrc	# Prettier config
└─ README.md	# Project documentation

## Key Design Principles Applied:

### 1. Feature-Based Architecture

- Each feature (accounts, transactions, budgets) has its own folder with hooks, utils, and types
- Promotes modularity and easier maintenance

## 2. Component Hierarchy

- **ui/** - Basic reusable components (Button, Input, etc.)
- **forms/** - Specialized form components
- **charts/** - Data visualization components
- **lists/** - List item components
- **layout/** - Layout and container components

## 3. Screen Organization

- Each screen has its own folder with related components
- Screen-specific components are co-located for better organization

## 4. State Management Structure

- RTK Query for data fetching and caching
- Feature slices for local state
- Selectors for derived state
- Clear separation between API and UI state

## 5. Service Layer

- Clean separation of concerns
- Reusable services across features
- Platform-specific implementations

## 6. Utility Organization

- Constants for design system values
- Helpers for common operations
- Custom hooks for reusable logic

## Memory-Friendly Development Strategy:

When the context gets full, you can ask for specific folders/files by referencing:

- **src/screens/[feature]/** - For specific screen implementations
- **src/components/[category]/** - For specific component types
- **src/features/[feature]/** - For feature-specific logic
- **src/services/[service]/** - For specific services

This structure ensures each part is self-contained and can be developed independently!