

Task Management Mobile Application â Project Summary

Overview

This project is a mobile task management application developed using **React Native (Expo)** with **Supabase** as the backend. The application enables users to create, view, edit, and delete tasks while maintaining secure authentication and session management.

The system follows a full-stack architecture, integrating frontend UI components with backend database operations and authentication services.

Technologies Used

Frontend

- React Native (Expo)
- Expo Router (navigation)
- React Hooks ('useState', 'useEffect', 'useFocusEffect')

Backend

- Supabase (PostgreSQL database + authentication)
- Row Level Security (RLS) for data protection

Storage & Security

- Expo SecureStore for persistent session storage

Application Structure

Navigation

The app uses **Expo Router** with a combination of:

- **Tab navigation** for main screens
- **Stack navigation** for individual pages

Main routes:

- '/' â Home (Today view)
- '/tasks' â Task list
- '/createTask' â Create new task
- '/editTask' â Edit existing task
- '/createUser' â Sign up
- '/login' â Login

Authentication System

The application includes a complete authentication flow:

- **User Registration**
 - Email and password-based signup
- **User Login**
 - Credential-based authentication
- **Session Management**
 - Persistent sessions using SecureStore
- **Protected Routes**
 - Users are redirected if not authenticated
- **Logout**
 - Ends session via Supabase

Task Management Features

Create Task

Users can:

- Enter title, description, and deadline
- Set completion status using a checkbox
- Save tasks to the database

Read Tasks

- Tasks are fetched from Supabase
- Displayed using a 'SectionList'
- Categorized into:
 - Upcoming Tasks
 - Completed Tasks
- Supports manual refresh and auto-refresh on screen focus

Update Task

- Existing tasks can be edited
- Fields are pre-filled with current values
- Updates are sent to the database using task ID

Delete Task

- Tasks can be deleted with confirmation
- List updates after deletion

User Interface Design

A centralized styling system ('defaultStyles') was implemented to ensure consistency across the application.

Key UI Components

- Text inputs for forms
- Buttons and pressable elements
- Custom checkbox component
- Sectioned task list
- Activity indicators for loading states

UX Features

- Keyboard handling with 'KeyboardAvoidingView'
- Tap outside to dismiss keyboard
- Alerts for feedback (success/errors)
- Dynamic headers with actions (refresh, logout)

State Management

The app uses local state via React Hooks:

- 'useState' for form data and UI state
- 'useEffect' for lifecycle events
- 'useFocusEffect' for refreshing data when screens are focused

Backend Integration

Supabase Client

- Configured using environment variables
- Secure session persistence enabled
- Automatic token refresh

Database Operations

- 'INSERT' â create tasks
- 'SELECT' â fetch tasks
- 'UPDATE' â edit tasks
- 'DELETE' â remove tasks

Data Model (Tasks Table)

- 'tId' (UUID)
- 'title'
- 'description'
- 'deadline'
- 'isCompleted'
- 'lastChanged'
- 'uId' (user reference)

Security

- Row Level Security (RLS) ensures users can only access their own tasks
- Queries are filtered using 'auth.uid() = uid'
- Update operations require a 'WHERE' clause to prevent unintended changes

Challenges and Solutions

Routing Issues

- Fixed incorrect relative paths by using absolute routes and parameters

Data Binding

- Ensured edit forms are pre-filled by fetching data using task ID

Database Errors

- Resolved missing 'WHERE' clause in update queries
- Handled invalid date formats

UI Layout Problems

- Improved header layout by replacing default buttons with custom components
- Fixed spacing and alignment issues

Conclusion

The project successfully demonstrates the development of a full-stack mobile application with:

- Secure authentication
- Persistent user sessions
- CRUD operations with a backend database
- Structured navigation and UI design

The application follows scalable patterns and provides a solid foundation for further enhancements such as improved UI design, additional features, and performance optimizations.