SevenApps React Native Case Study

Video Diary App

Objective
Develop a **React Native Video Diary App** where users can:
• Import videos,
• Crop a specific segment of 5 seconds,
• Add details such as name and description,
• Save cropped videos to a list for future reference.

The app should prioritize simplicity, efficiency, and scalability, adhering to modern React Native development practices.

Case Study Requirements

Main Features

1. **Main Screen: Cropped Video List**
   • Display a list of previously cropped videos.
   • Implement persistent storage (e.g., Zustand with AsyncStorage or an alternative state management solution).
   • Allow users to tap a video in the list to navigate to the **Details Page**.
2. **Details Page**
   • Display the selected video with its:
   • Name,
   • Description.
   • Keep the UI minimalistic, focusing on the video and its associated metadata.
3. **Crop Modal**
   • **Step 1:** Video Selection
   • Allow users to select a video from their device.

• **Step 2:** Video Cropping
• Display the video with a scrubber allowing users to select the start and end points for a 5-second segment.
• Below the scrubber, include a button to proceed to the next step.

• **Step 3:** Add Metadata
• Include two input fields:
• Name (text input),
• Description (text area).
• Add a button to execute the cropping operation.

4. **Video Cropping**
   • Implement video cropping functionality using [expo-trim-video].
   • The trimVideo function under expo-trim-video script should execute via **Tanstack Query**, ensuring asynchronous operations and a robust API integration.

Bonus Features

1. **Edit Page (Optional)**
   • Add a page allowing users to edit the **Name** and **Description** of a cropped video.
   • Include a form with two inputs:
   • **Name**
   • **Description**
   • Persist updates to storage.
2. **Enhancements**
   • Use **Expo SQLite** for storing and reading video data.
   • Integrate **React Native Reanimated** for smoother animations.
   • Use **Zod** or **Yup** for form input validation.

Required Technologies
• **Core Technologies**
• **Expo**: Base framework for React Native development.
• **Expo Router**: For implementing app navigation.
• **Zustand**: State management solution.
• **Tanstack Query**: To manage async logic and the expo-trim-video cropping process.
• **expo-trim-video**: Core library for video processing.
• **NativeWind**: Styling solution.
• **Expo Video**: Video rendering and playback (or any suitable alternative).

• **Bonus Technologies**
• **Expo SQLite**: For structured, persistent storage.
• **React Native Reanimated**: For animations.
• **Zod/Yup**: Validation schemas for form handling.

Key Considerations

1. **Scalability**
   • Ensure the app can handle a growing list of cropped videos efficiently.
   • Abstract key features into reusable components (e.g., VideoPlayer, MetadataForm).
2. **Performance**
   • Use efficient libraries like Tanstack Query to handle background processing.
3. **Usability**
   • Simplify navigation and UX design for intuitive user interactions.
   • Use NativeWind for clean and responsive styling.
4. **Validation**
   • Use Zod or Yup to validate user inputs for metadata (e.g., Name and Description).

Deliverables
• A functional React Native app with all specified features.
• Bonus features (if implemented) for enhanced functionality.
• Clean and modular codebase adhering to best practices.
• Documentation for setup and usage instructions.