Project Report: React Native To-Do Application

Abdulrhman Sobhy Hanafy

February 28, 2026

Executive Summary

This report outlines the development of a functional To-Do List application built to solidify cross-platform mobile development skills. The objective was to create a seamless, responsive task-management interface that runs smoothly on Android devices, built entirely utilizing React Native and Expo. The project demonstrates not only UI development and state management but also environment configuration and deep system troubleshooting.

Live Demonstration

The application code and a live interactive preview can be accessed and tested directly in the browser via Expo Snack:
Live Link: https://snack.expo.dev/@sobhys/to-do-list

Technical Stack

- **Framework:** React Native
- **Build Tool & Environment:** Expo
- **Development Environment:** Android Studio (Virtual Device Emulator)
- **System Tooling:** Windows PowerShell, Android Debug Bridge (ADB)

Core Features

- **Task Management:** Intuitive interface for users to quickly add new tasks to their daily list.
- **Dynamic State Handling:** Real-time User Interface updates when adding, completing, or removing items from the list using React hooks.
- **Cross-Platform Architecture:** Built with Expo, ensuring the underlying codebase is optimized for seamless compilation to both Android and iOS platforms.

Technical Challenges & Problem Solving

Building the UI and handling the React state is only one facet of mobile development; configuring the local build environment is equally critical.

During the build and deployment phase to the Android Studio emulator, a significant environment routing issue occurred. The Windows system environment variables were defaulting to a deprecated Genymotion path rather than the official Android Studio SDK location. This completely prevented the Android Debug Bridge (`adb`) from locating the SDK, breaking communication with the emulator.

The Resolution

Instead of relying on standard graphical interface fixes which can sometimes fail to apply system-wide, direct command-line intervention was used via PowerShell. The system environment variables were remapped as follows:

1. The `ANDROID_HOME` variable was re-routed to the correct local `AppData` directory: `C:\Users\Panda\AppData\Local\Android\Sdk`
2. The `platform-tools` and `emulator` paths were directly injected into the Windows `Path` variable.

This immediately restored `adb` functionality (verified via `adb --version`), allowing the Expo bundler to successfully compile the JavaScript bundles and launch the application seamlessly on the Android Studio emulator.

Future Enhancements

Moving forward, the roadmap for this application includes:

- **Persistent Storage:** Integrating local storage (such as `AsyncStorage`) so tasks remain saved after the application is closed or the device is rebooted.
- **Task Categorization:** Allowing users to create different lists (e.g., Work, Personal, Groceries).
- **Cloud Integration:** Exploring backend integrations (like Firebase) for cross-device syncing and user authentication.