Solana DApp Project Summary - Quick Interview Review

Tech Stack Overview

- React: UI library for building components.
- Vite: Fast development build tool.
- Solana Web3.js: JavaScript SDK to interact with Solana.
- @solana/wallet-adapter: Easily connect wallets like Phantom.
- bs58: Base58 encoding (used for signatures, addresses).

Components Summary

1. App.jsx: Main layout and wallet connection UI.
2. SignMessage.jsx: Allows signing of custom messages.
3. SendTokens.jsx: Sends SOL to another wallet address.
4. ShowSolBalance.jsx: Displays wallet balance.
5. RequestAirdrop.jsx: Requests test SOL from devnet.

Key Functions & Concepts

- useWallet(): Access publicKey, signMessage, sendTransaction.
- useConnection(): Access Solana connection.
- LAMPORTS_PER_SOL: 1 SOL = 1,000,000,000 lamports.
- SystemProgram.transfer(): Transfers SOL.
- new Transaction(): Creates blockchain transaction.

Important Interview Points

- Devnet is used for testing (not real SOL).
- Wallet Adapter helps connect frontend to Solana wallets.
- Avoid using getElementById in React - use state instead.

- Use Base58 (via bs58) to encode publicKey, signature.

Sample Explanation

"My app connects to the Solana devnet and uses wallet-adapter to allow users to sign in with a wallet like Phantom. It can sign messages, send SOL to other addresses, and request airdrops. React manages UI components, and Vite powers the frontend tooling."