Exploring Flutter State Management & Provider

By: Syed Murtaza Ali Shah

Key Takeaways from My Learning

- State management is crucial for handling dynamic UI updates in Flutter.
- Flutter offers various approaches like `setState`, `InheritedWidget`, `Provider`, and more.
- **Provider** is a scalable and easy-to-use solution for managing and sharing app state.
- It uses context to listen and update data reactively.

Real-World Use Case of Provider

E-Commerce App - Cart Management

- Manages cart state (add/remove items) globally.
- UI components like product tiles, cart icon, and checkout page share the same state.
- Ensures consistent updates and reactivity across the app

Questions & Confusions

- How does **Provider** compare in performance to **Riverpod** or **Bloc**?
- When should I use **ChangeNotifier** vs. **StateNotifier** or **Cubit**?
- How do I optimize performance in deeply nested widget trees?

The End