**CHAPTER 3**

I practiced referencing variables dynamically to update application states and explored Dart's various data types for different purposes, such as numbers for calculations, strings for text storage, booleans for managing states, lists for collections, maps for structured data, and runes for Unicode values. I also delved into Dart's operators for arithmetic, logical operations, and string concatenation.

In terms of flow control, I learned how to use if and else statements, the ternary operator, loops, and switch cases to implement logic like decision-making, UI updates, and retry mechanisms. Writing reusable functions proved essential for handling tasks such as saving entries, formatting dates, and refreshing the UI, making my code more efficient and maintainable. I explored importing packages like sqflite for database management, provider for state management, and intl for date and time formatting to extend functionality. Additionally, I gained hands-on experience with asynchronous programming using async and await, ensuring the app remained responsive during time-intensive operations such as database queries.

Overall, this guide provided a comprehensive introduction to Dart, equipping me with a strong foundation to effectively apply these concepts in real-world Flutter projects.