Sanlam Technical Test Submission

Candidate: Fanelesibonge Mbuyazi
Role Applied: Software Engineer I - Sanlam FinTech

This document provides the solution to the **Shopping Cart Implementation Exercise**, submitted as part of the Sanlam FinTech Software Engineer I technical test. The task was to improve an existing shopping cart code snippet for better readability, maintainability, efficiency, and correctness while preserving its core business logic.

Approach:
1. Refactored the code into a layered architecture (Controller, Service, Model, DTO).
2. Introduced strong typing with dedicated domain classes (`Cart`, `CartItem`).
3. Replaced *double* with *BigDecimal* for correct monetary calculations.
4. Extracted business logic into `ShoppingCartService` for testability and clarity.
5. Improved API design to follow REST best practices.
6. Added validation for input data and meaningful error responses.
7. Used immutable *record* classes for DTOs and data carriers.

Key Improvements:
- Cleaner, modular design with separation of concerns.
- Readable and maintainable project structure.
- Proper use of HTTP response codes (`201 Created`, `404 Not Found`, `400 Bad Request`).
- Extensible foundation for future features (remove items, discounts, persistence).
- In-memory cart management for simplicity (DB out of scope).

Project Architecture:
- **Controller Layer:** Handles HTTP requests and delegates to the service.
- **Service Layer:** Contains core business logic.
- **Model Layer:** Represents domain entities (Cart, CartItem).
- **DTO Layer:** Structures API responses (`CartResponse`).
- **Exception Layer:** Defines custom error handling (`CartNotFoundException`).

Notes:
- Security, persistence, and unit testing were excluded as per test instructions.
- Application runs with Spring Boot (Java 17+, Maven).

This solution reflects my approach to delivering clean, maintainable, and production-ready code that balances correctness with simplicity. **Submitted by:** Fanelesibonge Mbuyazi