Separation of Concerns

Dividing a program into distinct sections to improve maintainability, scalability, and modularity.

Benefits of Separation of Concerns

1. Enhanced Maintainability
2. Improved Scalability
3. Efficient Collaboration
4. Reusability
5. Testability

Data Abstraction

Hiding the complexity of data implementation and showing only the necessary features.

Data Encapsulation

Restricting direct access to data to protect it from unintended modification.

Difference: Abstraction vs Encapsulation

Abstraction focuses on *what* an object does, while Encapsulation focuses on *how* data is protected.

Data Types

Define what kind of values a variable can store (e.g., int, float, string, boolean).

Data Representation

How data is stored in memory (e.g., Integer in binary, Characters in ASCII).

Interface vs Implementation

An Interface defines what an object should do, whereas Implementation is the actual code execution.

Abstract Data Types (ADTs)

Concepts that define operations on data without specifying how they are implemented.

Examples of ADTs

1. List
2. Stack (LIFO)
3. Queue (FIFO)
4. Set
5. Dictionary (Key-Value pairs)

Advantages of ADTs

1. Data Independence
2. Modularity
3. Easier Code Maintenance
4. Better Performance

Case Study - ShopEase Inventory

Using Separation of Concerns, ADTs, and Encapsulation to design a scalable inventory system.