CMC UNIVERSITY

OOP introduction

PhD. Ngo Hoang Huy

Wednesday, June 28, 2023

CMC UNIVERSITY

Objectives

Abstraction, encapsulation, hierarchy, and polymorphism are the foundational concepts of object-oriented programming.

They empower developers to create modular, maintainable, and extensible applications that effectively model the real world and provide elegant solutions to complex problems.

OOP offers a robust framework for developing complex applications, emphasizing modular design, code reusability, and maintainability.

developers can create software systems that are easier to understand, modify, and extend, resulting in more efficient and scalable solutions.

CMC UNIVERSITY

OOP Language for Complex Applications

Many software applications are complex, involving intricate designs, extensive functionality, and underlying problem domains with intricate details and complexities.

Understanding and addressing these fundamental complexities is crucial for developing robust and effective software solutions.

Balancing the right amount of complexity within the application is essential for accurately addressing real-world challenges and ensuring practicality and usability.

CMC UNIVERSITY

Complexity

There are 02 approaches to determining the most important features: through activities (distinct algorithms) and entities (distinct objects).

These approaches are not mutually exclusive. Starting with one approach, the results serve as the foundation for the other approach.

The decomposition process is iterative, continuously refining the solutions.

By combining both approaches, the crucial features can be effectively identified and prioritized.

CMC UNIVERSITY

Complexity

Xét ví dụ một hệ thống quản lý tư viện dạng cơ bản, nó gồm các chức năng:

Đăng nhập: Người dùng và thủ thư có thể đăng nhập vào hệ thống bằng tài khoản của mình.

Quản lý người dùng:

+Đăng ký: Người dùng có thể đăng ký tài khoản mới.

+Xem thông tin cá nhân: Người dùng có thể xem và chỉnh sửa thông tin cá nhân của mình.

Quản lý sách:

+Tìm kiếm sách: Người dùng và thủ thư có thể tìm kiếm sách theo tiêu chí như tên sách, tác giả, hoặc thể loại.

+Xem thông tin chi tiết sách: Người dùng và thủ thư có thể xem thông tin chi tiết về một cuốn sách cụ thể.

+Mượn sách: Người dùng có thể mượn sách từ thư viện.

+Trả sách: Người dùng có thể trả sách đã mượn cho thư viện.

Quản lý giao dịch:

Lưu trữ giao dịch: Hệ thống lưu trữ thông tin về các giao dịch mượn/trả sách.

Kiểm tra lịch sử giao dịch: Người dùng có thể kiểm tra lịch sử giao dịch của mình.

Complexity

Quản lý nhân viên:

+Xem thông tin cá nhân: Thủ thư và nhân viên có thể xem và chỉnh sửa thông tin cá nhân của mình.

+Quản lý lịch làm việc: Thủ thư và nhân viên có thể quản lý lịch làm việc của mình.

Quản lý cơ sở dữ liệu:

+Lưu trữ thông tin sách: Hệ thống lưu trữ thông tin về sách, bao gồm tên sách, tác giả, thể loại, và số lượng có sẵn.

+Lưu trữ thông tin người dùng: Hệ thống lưu trữ thông tin cá nhân của người dùng, bao gồm tên, địa chỉ, và thông tin liên lạc.

Sơ đồ chức năng này giúp hiểu rõ các chức năng chính của hệ quản lý thư viện và mối quan hệ giữa chúng

Complexity

Một số nhược điểm của phân tích hướng chức năng trong hệ quản lý thư viện là:

Khó khăn trong việc mở rộng và thay đổi: Sơ đồ hướng chức năng có thể gặp khó khăn khi cần mở rộng hoặc thay đổi chức năng của hệ thống quản lý thư viện. Khi có yêu cầu mới hoặc thay đổi trong quy trình hoạt động, việc thay đổi sơ đồ chức năng có thể yêu cầu sửa đổi nhiều thành phần liên quan, gây ra sự rối loạn và rủi ro lỗi.

Sự phụ thuộc mạnh mẽ giữa các chức năng: Trong sơ đồ hướng chức năng, các chức năng thường được thiết kế và triển khai một cách phụ thuộc mạnh mẽ vào nhau. Điều này có thể làm cho hệ thống khó khăn để quản lý và bảo trì. Một thay đổi nhỏ trong một chức năng có thể ảnh hưởng đến các chức năng khác và yêu cầu sự điều chỉnh phức tạp.

Thiếu sự linh hoạt và tái sử dụng: Sơ đồ hướng chức năng có thể thiếu sự linh hoạt và khả năng tái sử dụng. Các chức năng được thiết kế để phục vụ một mục đích cụ thể và không dễ dàng tái sử dụng trong các tình huống khác. Điều này có thể gây lãng phí thời gian và công sức khi phát triển và bảo trì hệ thống.

CMC UNIVERSITY

Object orientd

Các lớp đối tượng giúp phân chia và quản lý các nghiệp vụ cụ thể trong hệ thống quản lý thư viện.

Trong mô hình hướng đối tượng, khối chính thường đại diện cho các đối tượng khác nhau trong hệ thống, các thuộc tính của chúng, các chức năng khác nhau và các mối quan hệ giữa các đối tượng. Những khối xây dựng này được gọi là Sơ đồ lớp.

Lớp User: Quản lý các hoạt động của người dùng, bao gồm đăng nhập, đăng ký, xem thông tin cá nhân, và tìm kiếm sách.

Lớp Librarian: Quản lý các hoạt động của thủ thư, bao gồm thêm sách vào thư viện, cho mượn sách, nhận sách trả lại, và quản lý thông tin sách.

Lớp Book: Quản lý các hoạt động liên quan đến sách, bao gồm tìm kiếm sách, đặt sách, xem thông tin chi tiết về sách, và kiểm tra số lượng sách còn trong thư viện.

Lớp Account: Quản lý các hoạt động liên quan đến tài khoản người dùng, bao gồm tạo tài khoản, cập nhật thông tin cá nhân, và đổi mật khẩu.

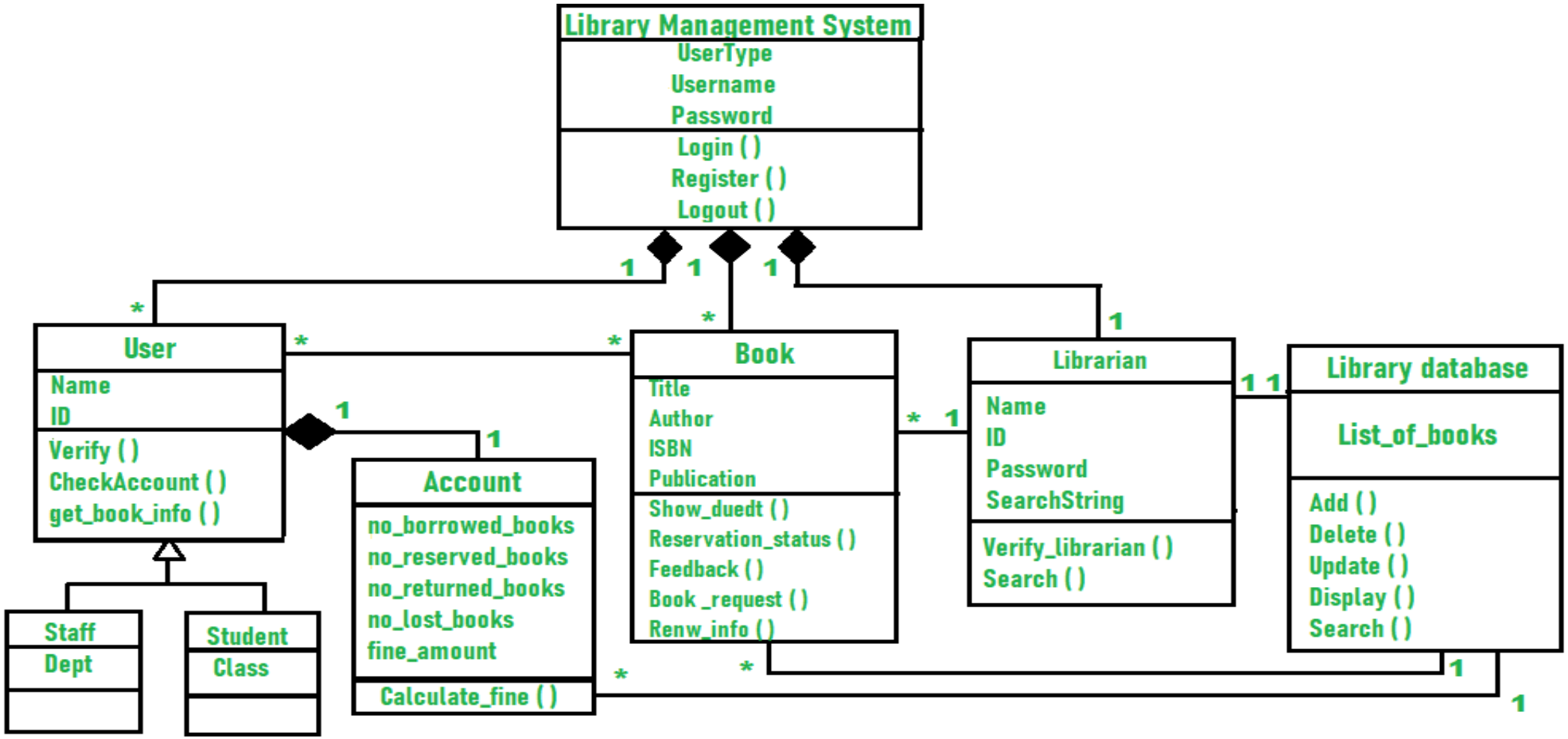
Lớp Library Database: Quản lý các hoạt động liên quan đến cơ sở dữ liệu thư viện, bao gồm lưu trữ thông tin về sách, người dùng, và các giao dịch mượn/trả sách.

Lớp Staff: Quản lý các hoạt động liên quan đến nhân viên, bao gồm quản lý thông tin cá nhân, lịch làm việc, và bảng lương.

Lớp Student: Quản lý các hoạt động liên quan đến sinh viên, bao gồm xem thông tin cá nhân, đăng ký mượn sách, và kiểm tra lịch sử giao dịch.

CMC UNIVERSITY

Object oriented



CLASS DIAGRAM FOR LIBRARY MANAGEMENT SYSTEM

Object oriented

- Với hướng đối tượng, bạn có thể tạo ra các đối tượng độc lập và tái sử dụng, và quản lý mối quan hệ giữa các đối tượng của hệ thống, tạo ra một hệ thống linh hoạt, dễ mở rộng và dễ bảo trì.
- Sử dụng kế thừa, đa hình, ta xác định các quan hệ giữa các đối tượng. Ví dụ, trong hệ thống quản lý thư viện, có các đối tượng Sách, Người dùng và Thư viện, và có quan hệ "một-nhiều" hoặc "nhiều-nhiều".
- Hướng đối tượng cung cấp mô hình linh hoạt và hiệu quả cho thiết kế và bảo trì hệ thống quản lý thư viện.

Object Terminology

04 fundamental concepts

- Encapsulation
- Inherit (or Hierarchy)
- Polymorphism
- Abstraction

CMC UNIVERSITY

Encapsulation

Lớp "Book" có thể có các thuộc tính như "tên sách", "tác giả", "năm xuất bản" và "số lượng hiện có".

Các thuộc tính này được bao gói trong lớp "Book" và không thể truy cập trực tiếp từ bên ngoài lớp. Thay vào đó, chúng ta có thể sử dụng các phương thức như "lấy thông tin sách", "cập nhật số lượng sách" để truy cập và thay đổi giá trị của các thuộc tính.

Lớp "Library" có thể có các thuộc tính như "tên thư viện", "địa chỉ" và "danh sách sách trong thư viện".

Các thuộc tính này cũng được bao gói trong lớp "Library" và không thể truy cập trực tiếp từ bên ngoài. Thay vào đó, chúng ta có thể sử dụng các phương thức như "thêm sách vào thư viện", "xóa sách khỏi thư viện" để thực hiện các thao tác liên quan đến sách trong thư viện.

Qua việc sử dụng Encapsulation, các thuộc tính của lớp "Book" và "Library" được bảo vệ và chỉ có thể truy cập thông qua các phương thức đã được xác định. Điều này giúp giữ gìn tính toàn vẹn dữ liệu và đảm bảo sự ổn định của hệ thống quản lý thư viện.

CMC UNIVERSITY

Inherit

Chúng ta có thể sử dụng khái niệm Inheritance (Kế thừa) trong các lớp như "User" (Người dùng) và "Librarian" (Thủ thư).

Lớp "User" có các thuộc tính và phương thức chung cho tất cả các người dùng trong hệ thống quản lý thư viện, ví dụ như "tên người dùng" và "địa chỉ email".

Chúng ta có thể tạo một lớp con của "User" là "Student" (Sinh viên), có thêm thuộc tính và phương thức riêng như "mã sinh viên" và "xem danh sách sách đã mượn".

Lớp "Librarian" có các thuộc tính và phương thức đặc thù cho công việc của thủ thư trong hệ thống quản lý thư viện, ví dụ như "số lượng sách đang được mượn" và "thêm sách vào thư viện".

Chúng ta có thể tạo một lớp con của "Librarian" là "HeadLibrarian" (Thủ thư trưởng), có thêm thuộc tính và phương thức riêng như "quản lý danh sách thủ thư" và "xem thống kê số lượng sách mượn".

Khi sử dụng kế thừa, lớp con sẽ kế thừa các thuộc tính và phương thức của lớp cha. Điều này giúp giảm sự lặp lại mã nguồn và tăng tính tái sử dụng trong hệ thống quản lý thư viện.

CMC UNIVERSITY

Polymorphism

Trong hệ quản lý thư viện, chúng ta có thể áp dụng khái niệm Polymorphism (Đa hình) trong các lớp như "Item" (Mục sách) và "User" (Người dùng).

Lớp "Item" có phương thức "displayInfo()" để hiển thị thông tin chi tiết về một mục sách trong hệ thống.

Chúng ta có thể tạo các lớp con của "Item" như "Book" (Sách) và "Magazine" (Tạp chí), mỗi lớp con sẽ triển khai phương thức "displayInfo()" theo cách riêng để hiển thị thông tin đặc thù của từng loại mục sách.

Lớp "User" có phương thức "borrowItem(Item item)" để mượn một mục sách từ thư viện.

Chúng ta có thể tạo các lớp con của "User" như "Student" (Sinh viên) và "Faculty" (Giảng viên), mỗi lớp con sẽ triển khai phương thức "borrowItem()" theo cách riêng để xử lý quyền hạn và điều kiện mượn sách tương ứng với từng nhóm người dùng.

Sử dụng đa hình, chúng ta có thể xử lý các đối tượng của các lớp con một cách đồng nhất thông qua các phương thức được định nghĩa trong lớp cha. Điều này giúp giảm sự phức tạp trong việc xử lý các đối tượng khác nhau và tăng tính linh hoạt của hệ quản lý thư viện.

CMC UNIVERSITY

Abstraction

Trong hệ quản lý thư viện, chúng ta có thể áp dụng khái niệm Abstraction (Trừu tượng) trong các lớp như "Library" (Thư viện) và "Item" (Mục sách).

Lớp "Library" có phương thức "searchItem(String keyword)" để tìm kiếm mục sách trong hệ thống.

Thay vì tiết lộ chi tiết cụ thể về cách tìm kiếm và lưu trữ, chúng ta chỉ cần tập trung vào chức năng chính của việc tìm kiếm mục sách. Điều này tạo ra một mức độ trừu tượng, giúp chúng ta xây dựng một giao diện đơn giản và dễ sử dụng cho người dùng.

Lớp "Item" có thuộc tính "title" (tiêu đề) và phương thức "displayInfo()" để hiển thị thông tin chi tiết về một mục sách trong hệ thống.

Chúng ta không quan tâm đến các chi tiết cụ thể về việc lưu trữ dữ liệu hay các thuộc tính khác của mục sách. Chỉ cần tập trung vào việc hiển thị thông tin cơ bản và trừu tượng hóa các chi tiết phức tạp.

Sử dụng trừu tượng, chúng ta giảm bớt sự phức tạp và chi tiết không cần thiết, tạo ra các lớp và giao diện đơn giản, dễ sử dụng và dễ bảo trì. Trừu tượng giúp chúng ta tách biệt khái niệm chức năng và chi tiết triển khai, tạo ra một mức độ trừu tượng cao hơn, tập trung vào các khái niệm quan trọng và tạo ra sự rõ ràng và linh hoạt trong hệ quản lý thư viện.

CMC UNIVERSITY

THANK YOU