



OOP PROJECT

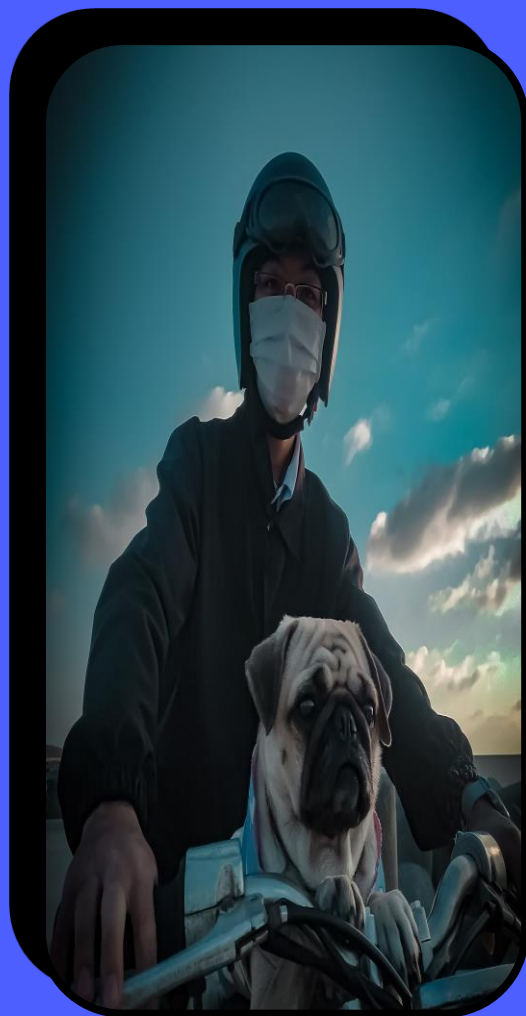
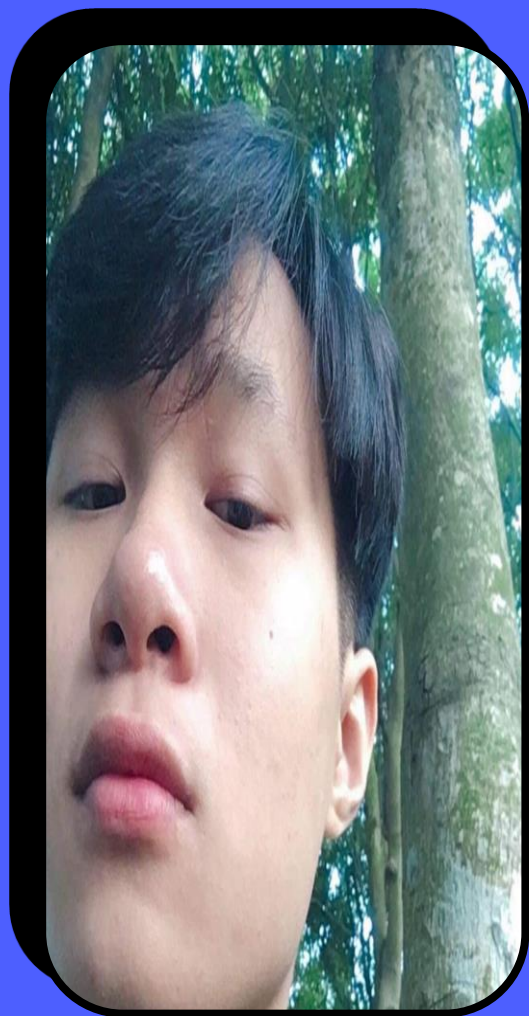


BUILDER DESIGN PATTERN



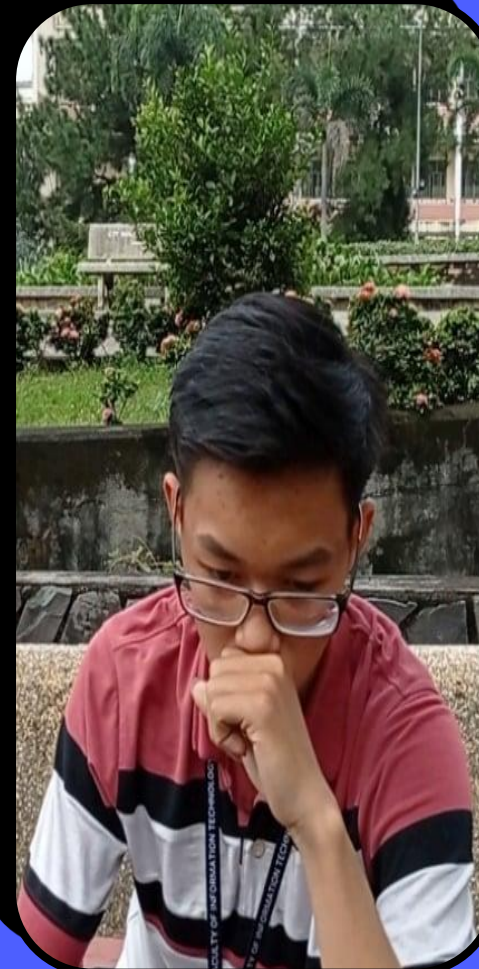
GROUP 6

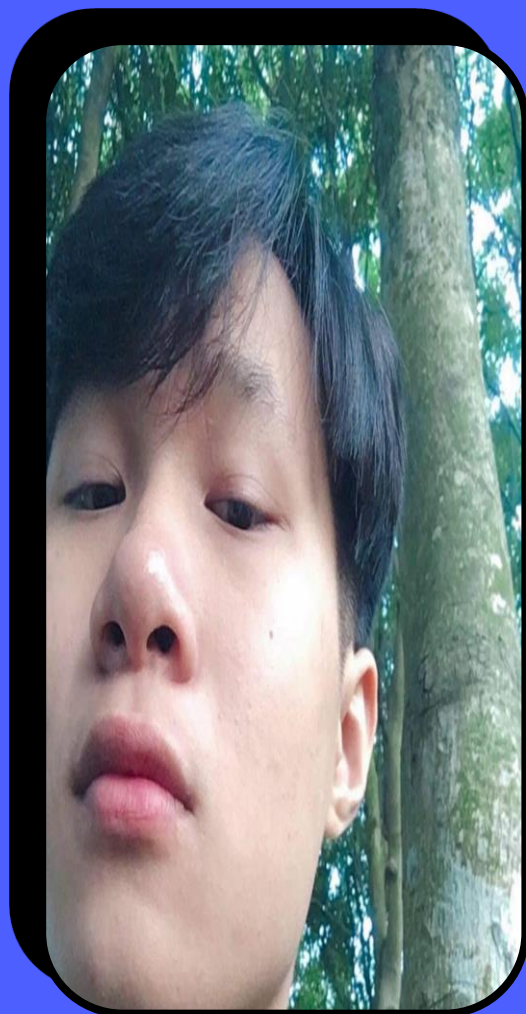




Nguyễn Quốc Tuấn
20127659

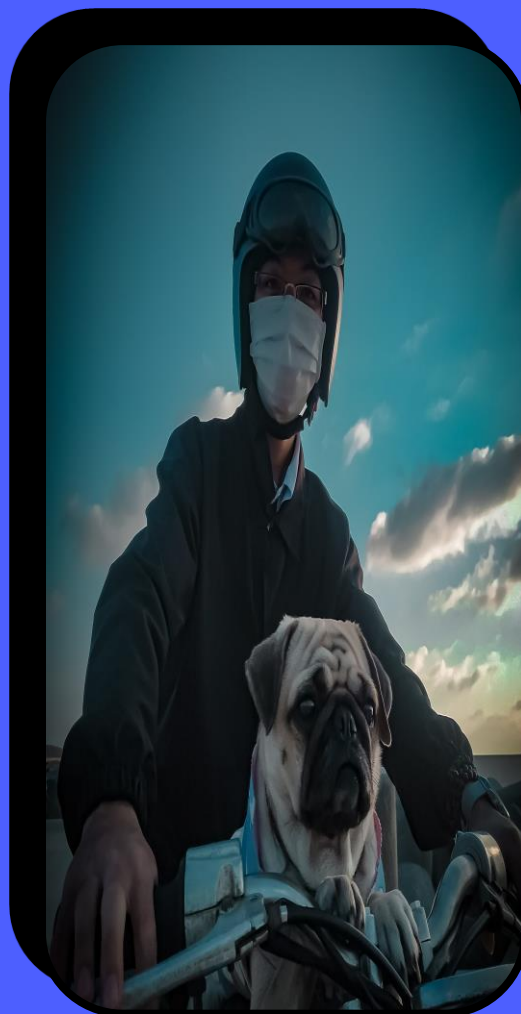
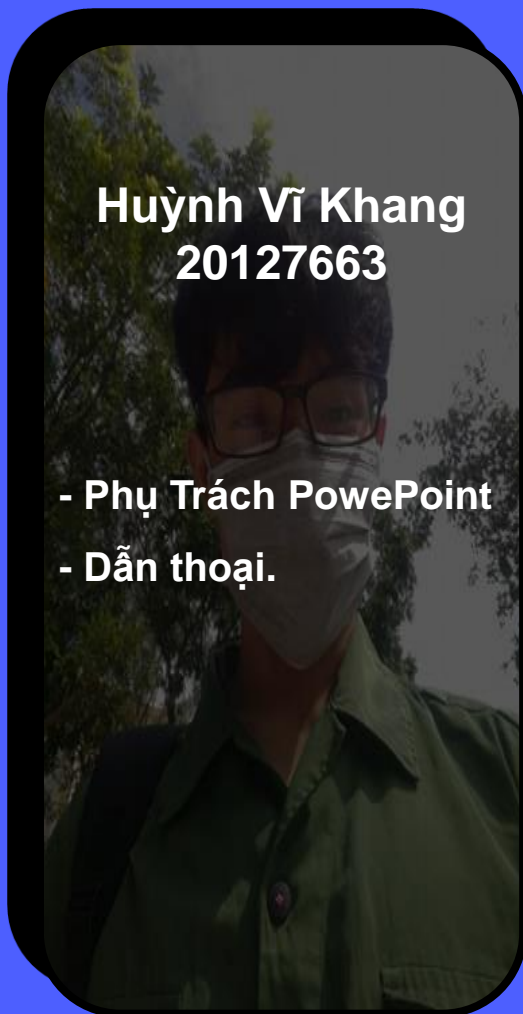
- Trưởng nhóm
- Dẫn thoại
- Demo code

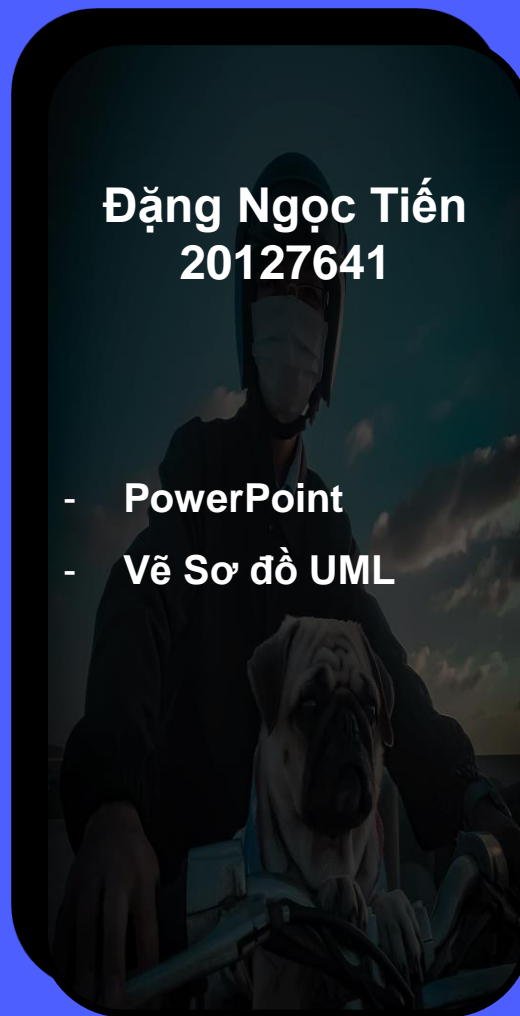
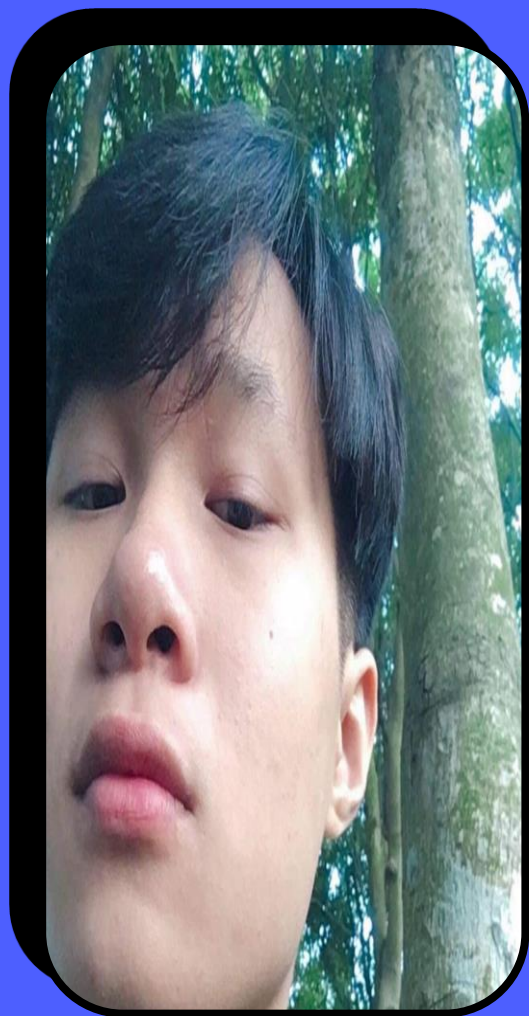




Huỳnh Vĩ Khang
20127663

- Phụ Trách PowerPoint
- Dẫn thoại.

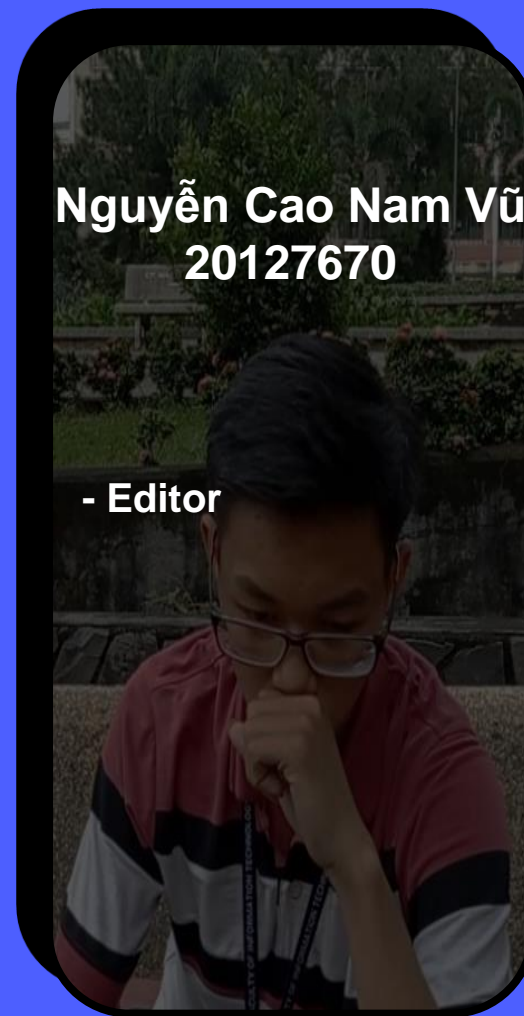
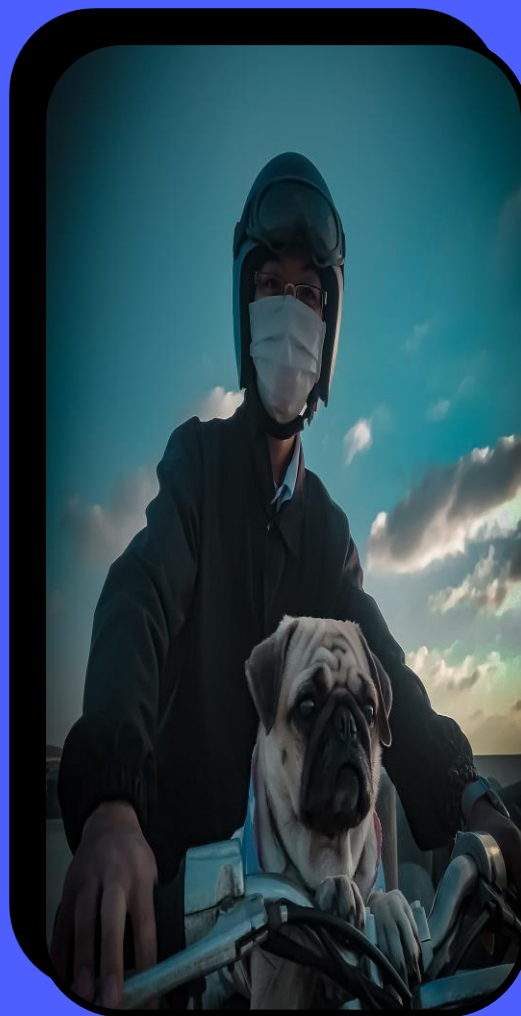
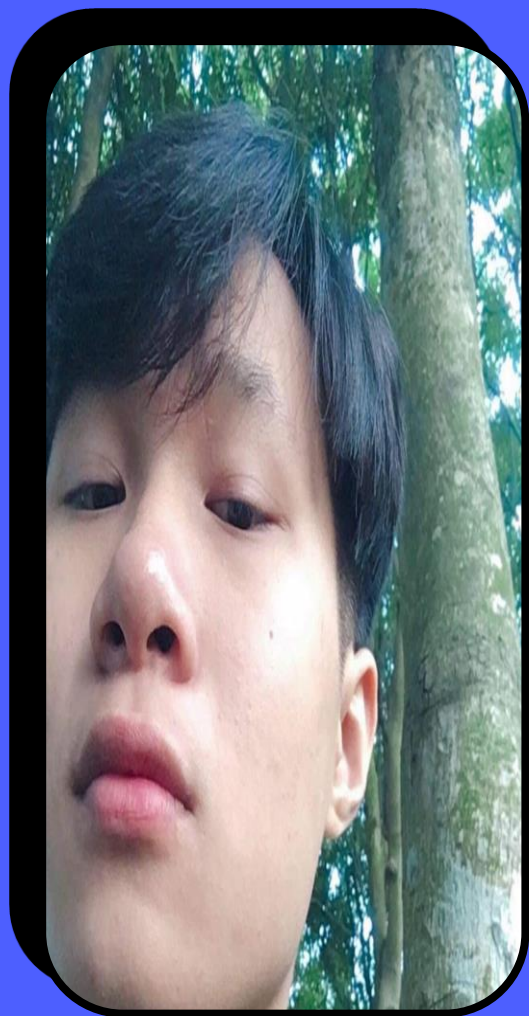




Đặng Ngọc Tiến
20127641

- PowerPoint
- Vẽ Sơ đồ UML







BẢNG PHÂN CÔNG CÔNG VIỆC

TUẤN
32.5%

KHANG
25%

TIẾN
22.5%

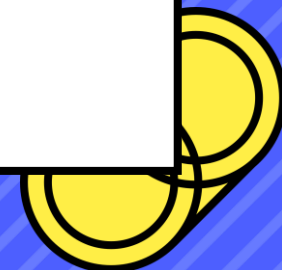
VŨ
20%

- Tổng hợp nội dung để thuyết trình.
- Lên kế hoạch cho phần code của Builder Pattern.
- Lên ý tưởng chính cho code.
- Dẫn thoại cho demo code.

- Lên kế hoạch và làm PPT cho các mục: khái niệm, khi nào và cách sử dụng builder
- Dẫn thoại cho các phần lý thuyết.

- Phụ trách vẽ UML cho code
- Làm PPT cho các mục: Sơ đồ UML, ưu nhược điểm của builder

- Tìm hiểu các phần lý thuyết cần thiết
- Edit video



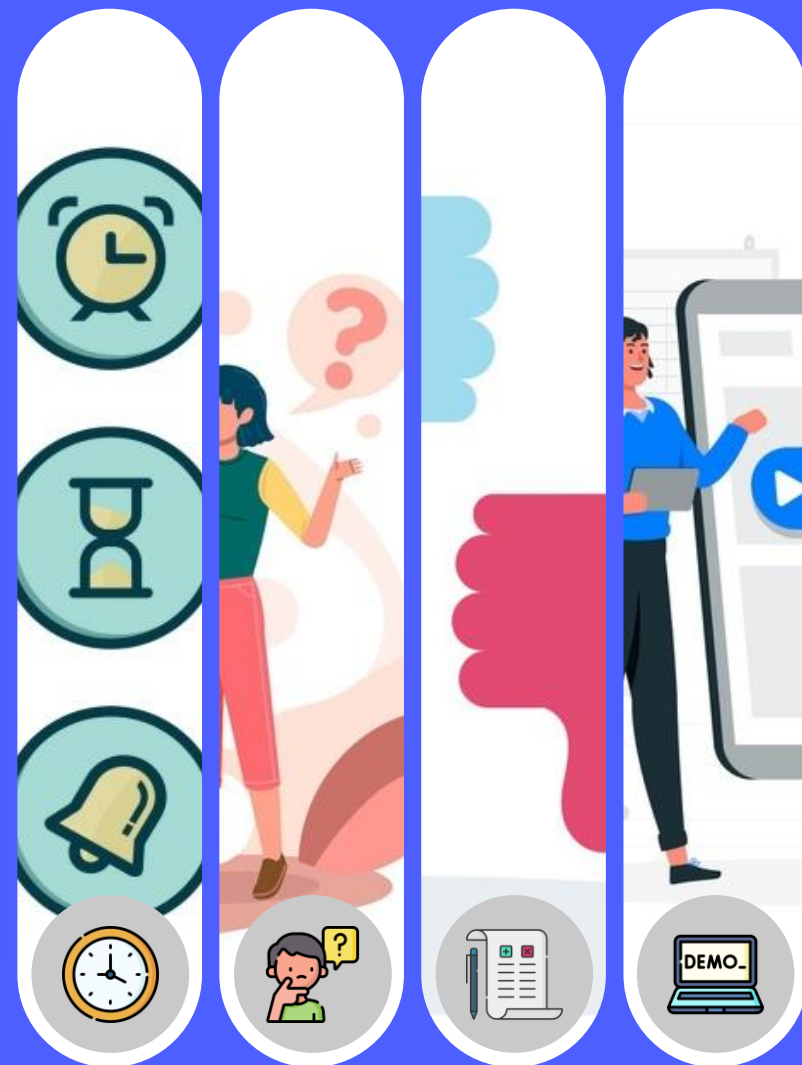


Lets begin

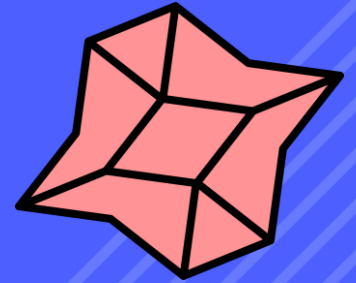


KHÁI NIỆM BUILDER DESIGN PATTERN.

Builder Design Pattern xây dựng 1 đối tượng phức tạp từ nhiều các đối tượng đơn giản từng bước một. Quá trình xây dựng lên khung sườn của đối tượng phức tạp nên được xây dựng 1 cách tổng quan, chung chung. Để các đối tượng phức tạp tương tự có thể được xây dựng dựa trên khung sườn của đối tượng phức tạp ban đầu đó.



Một số ví dụ

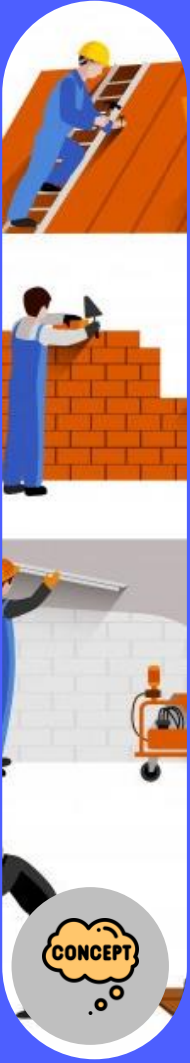


Nhà có thể tạo thành các đối tượng phức tạp khác nhau như nhà, villa, căn hộ nhỏ,...

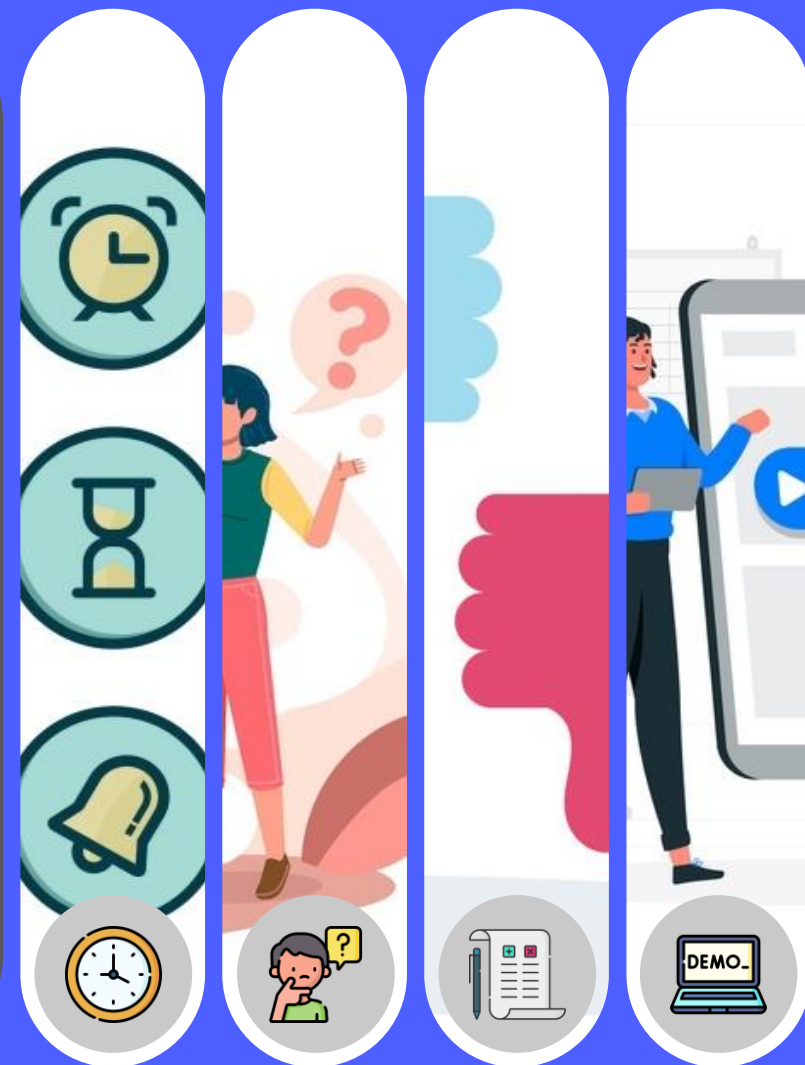
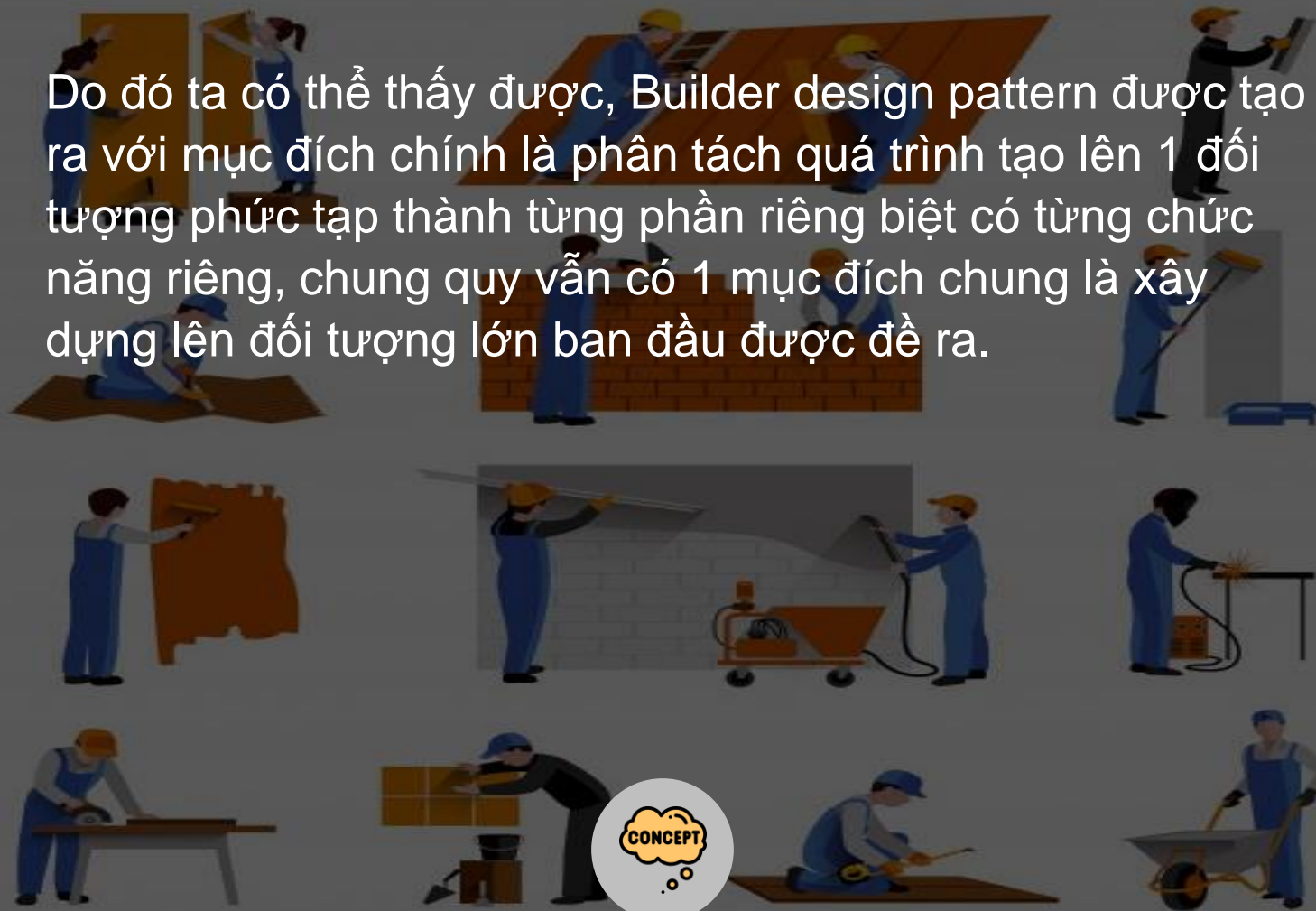


Cà phê cũng có thể tính là 1 đối tượng phức tạp với nhiều thuộc tính khác nhau,...



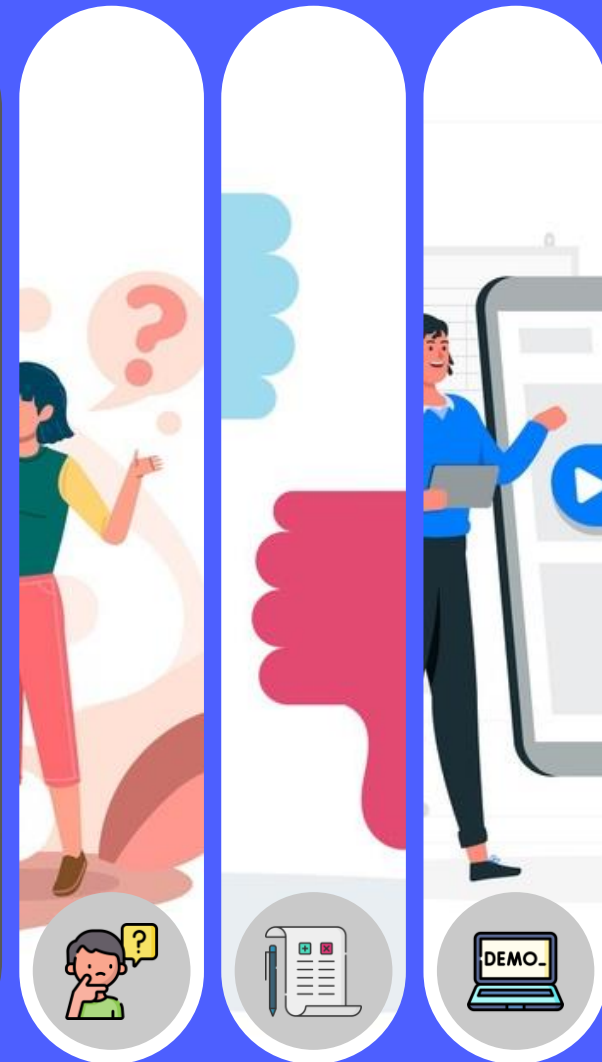


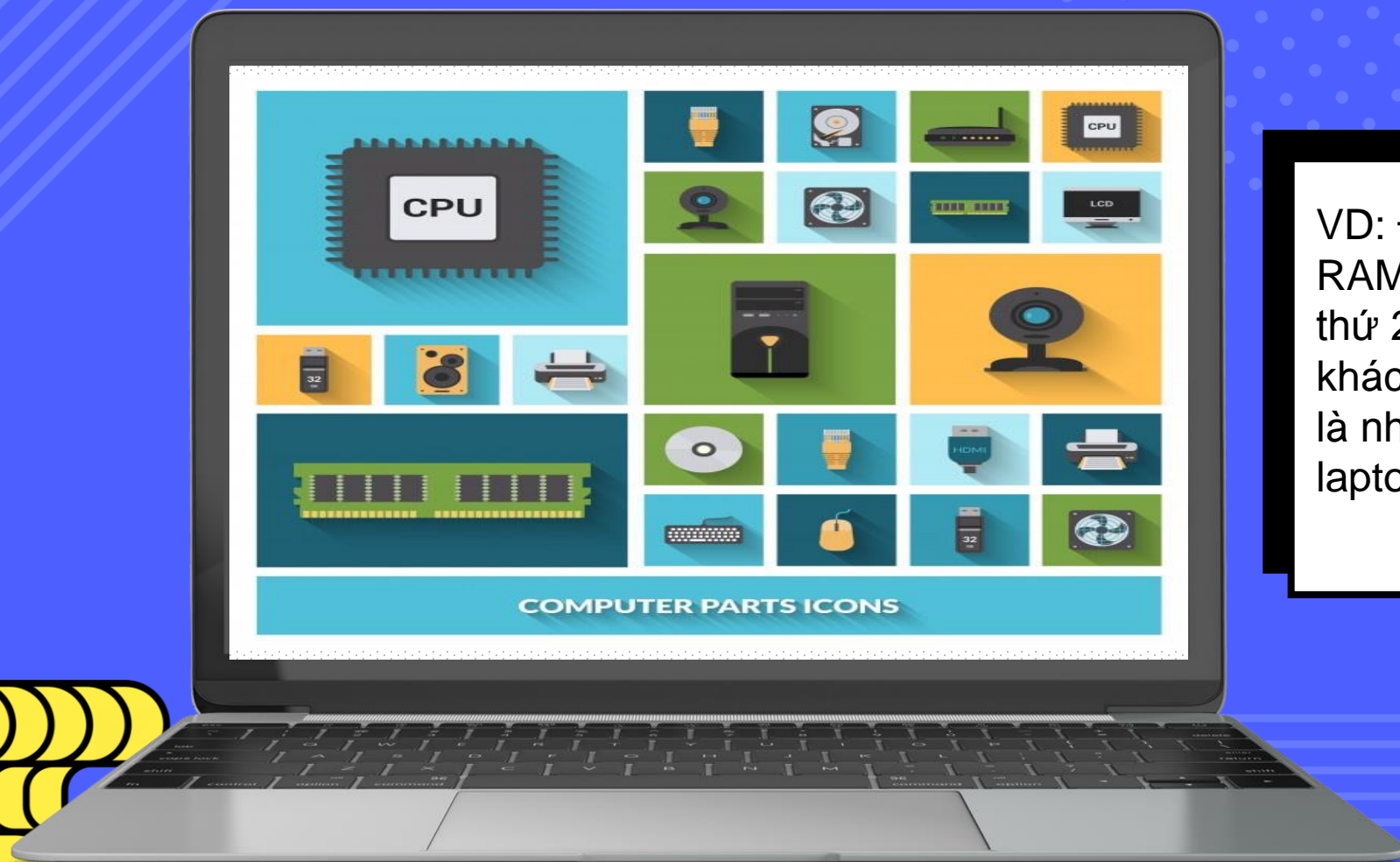
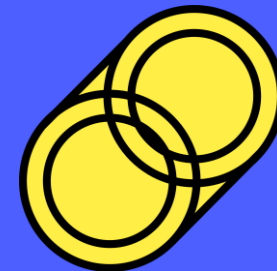
Do đó ta có thể thấy được, Builder design pattern được tạo ra với mục đích chính là phân tách quá trình tạo lên 1 đối tượng phức tạp thành từng phần riêng biệt có từng chức năng riêng, chung quy vẫn có 1 mục đích chung là xây dựng lên đối tượng lớn ban đầu được đề ra.



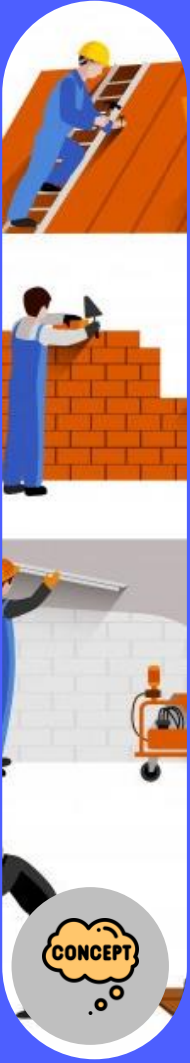
KHI NÀO CHÚNG TA NÊN SỬ DỤNG BUILDER.

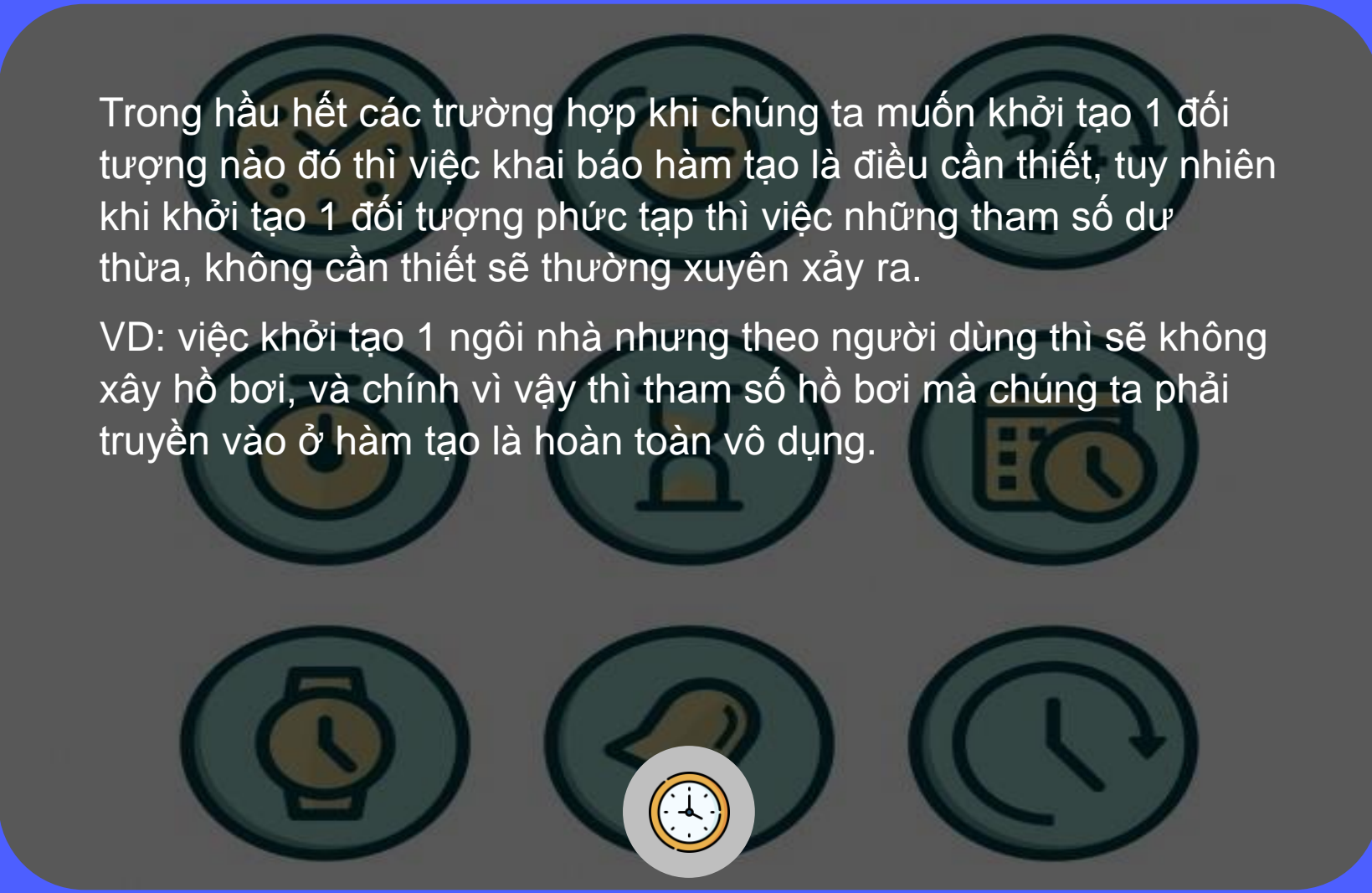
-Builder pattern là 1 phương pháp được sử dụng khá rộng rãi trong việc lập trình hướng đối tượng C++. Đặc biệt hiệu quả khi chúng ta cần tạo ra những đối tượng cần chỉnh sửa nhiều về thuộc tính nhỏ của nó.





VD: Đối tượng laptop cần thay thành RAM mới hoặc là thêm vào thành RAM thứ 2, laptop cũng có thể thay ổ cứng khác. Ở đây thì RAM và ổ cứng chính là những đối tượng nhỏ của đối tượng laptop phức tạp.



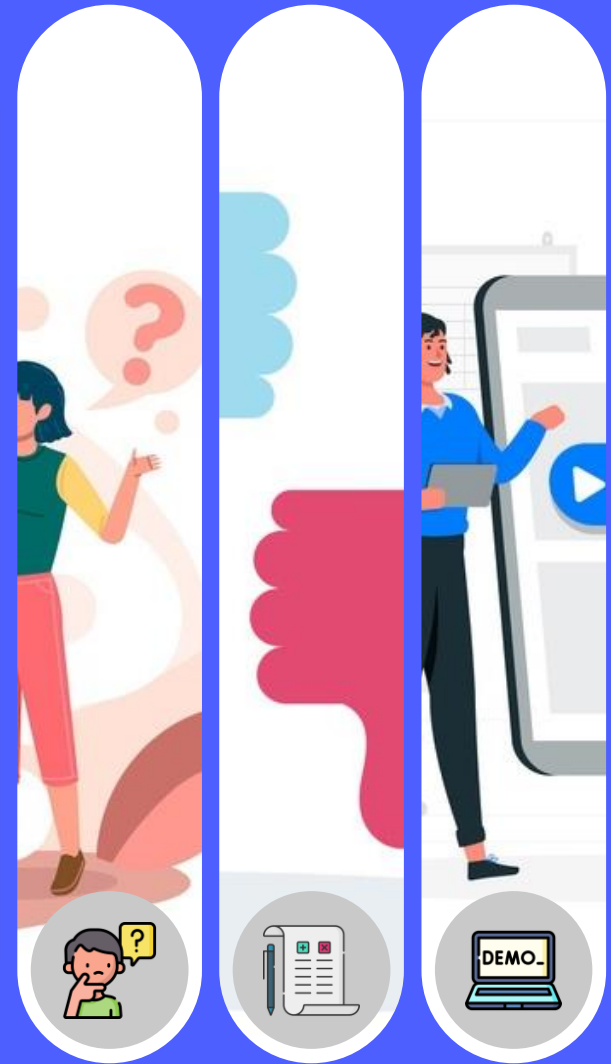


Trong hầu hết các trường hợp khi chúng ta muốn khởi tạo 1 đối tượng nào đó thì việc khai báo hàm tạo là điều cần thiết, tuy nhiên khi khởi tạo 1 đối tượng phức tạp thì việc những tham số dư thừa, không cần thiết sẽ thường xuyên xảy ra.

VD: việc khởi tạo 1 ngôi nhà nhưng theo người dùng thì sẽ không xây hồ bơi, và chính vì vậy thì tham số hồ bơi mà chúng ta phải truyền vào ở hàm tạo là hoàn toàn vô dụng.



CONCEPT



TELESCOPING CONSTRUCTOR.

Giả sử chúng ta có 1 class rất nhiều thuộc tính và chúng có thể kết hợp với nhau để khởi tạo 1 object theo nhiều cách. Nghĩa là chúng ta có 1 số tổ hợp các thuộc tính để tạo ra các constructor khác nhau, nếu không khéo sẽ khiến code bị trùng lặp và khó quản lí.

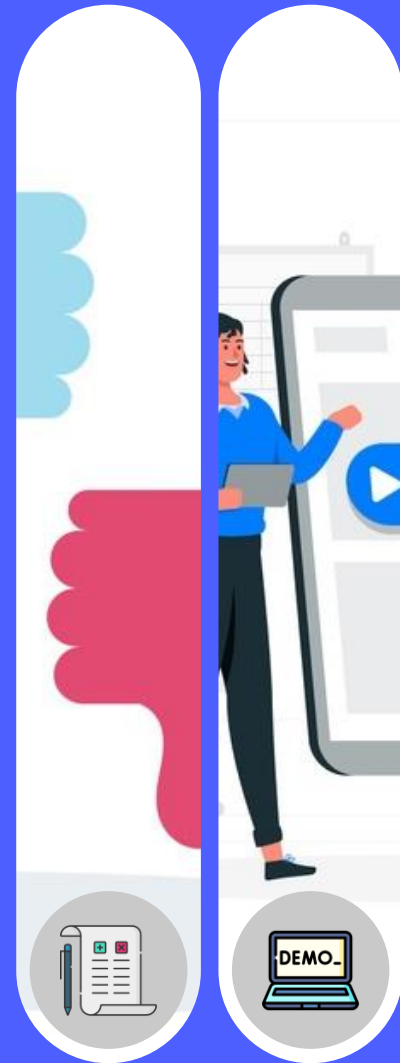
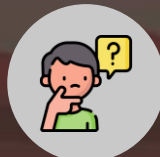




CÁCH SỬ DỤNG BUILDER DESIGN PATTERN:

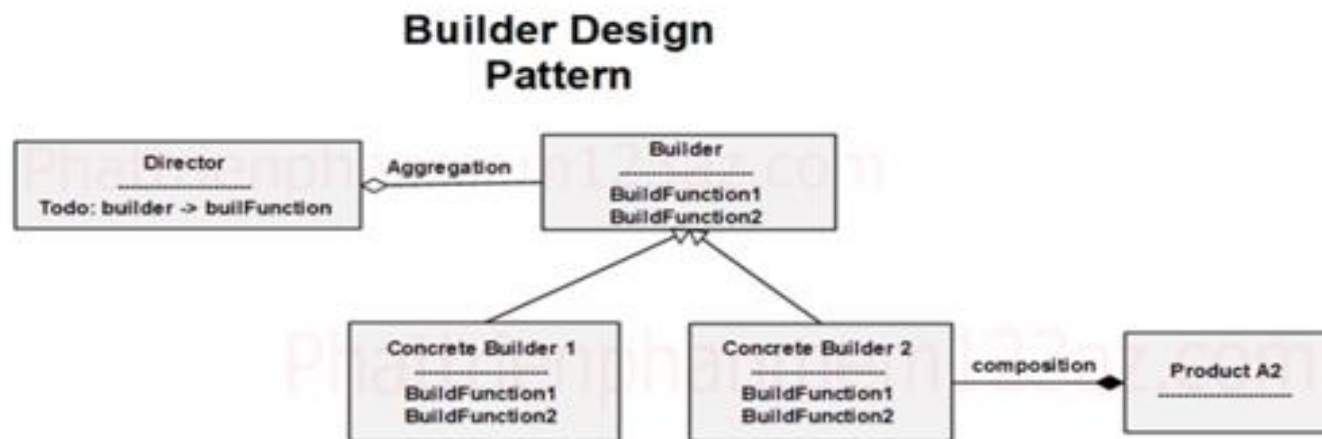
Khi sử dụng Builder Pattern chúng ta sẽ phân tách code dùng trong việc khởi tạo (VD: khởi tạo cửa chính, khởi tạo cửa sổ, tường,...) của đối tượng phức tạp mà chúng ta nhắm đến (ở đây là House) thành 1 lớp hoàn toàn tách biệt gọi là Builder.

Khung sườn cho việc khởi tạo đối tượng phức tạp này nên được xây dựng 1 cách tổng quan, chung chung. Để các đối tượng phức tạp tương tự có thể được xây dựng dựa trên khung sườn của đối tượng phức tạp ban đầu đó.



Mẫu Builder chung.

A GENERAL UML OF BUILER PATTERN





GIỚI THIỆU UML



Product

Lớp này sẽ định nghĩa loại của đối tượng phức tạp chúng ta chuẩn bị hướng đến.



Builder

Đây sẽ là lớp trừu tượng định nghĩa cho toàn bộ hành vi của quá trình xây dựng nên đối tượng phức tạp mà chúng ta nhắm đến.



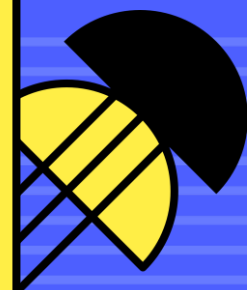
Concrete Builder

Đây sẽ là thành phần triển khai, cụ thể hóa các lớp trừu tượng để tạo ra các thành phần và tập hợp các thành phần đó với nhau.



Director

Lớp này chịu trách nhiệm chính về việc điều khiển tất cả các thuật toán để tạo thành sản phẩm cuối cùng.





GIỚI THIỆU UML



Product

Lớp này sẽ định nghĩa loại của đối tượng phức tạp chúng ta chuẩn bị hướng đến.



Builder

Đây sẽ là lớp trừu tượng định nghĩa cho toàn bộ hành vi của quá trình xây dựng nên đối tượng phức tạp mà chúng ta nhắm đến.



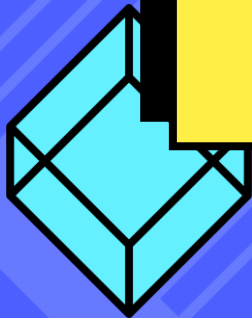
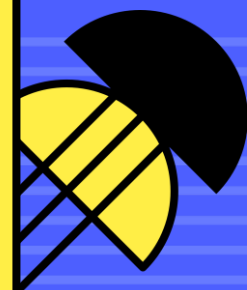
Concrete Builder

Đây sẽ là thành phần triển khai, cụ thể hóa các lớp trừu tượng để tạo ra các thành phần và tập hợp các thành phần đó với nhau.



Director

Lớp này chịu trách nhiệm chính về việc điều khiển tất cả các thuật toán để tạo thành sản phẩm cuối cùng.





GIỚI THIỆU UML



Product

Lớp này sẽ định nghĩa loại của đối tượng phức tạp chúng ta chuẩn bị hướng đến.



Builder

Đây sẽ là lớp trừu tượng định nghĩa cho toàn bộ hành vi của quá trình xây dựng nên đối tượng phức tạp mà chúng ta nhắm đến.



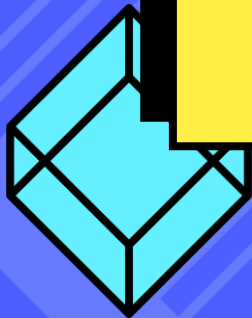
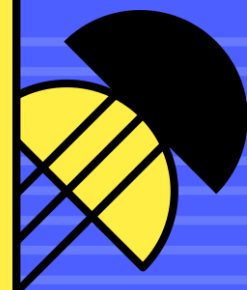
Concrete Builder

Đây sẽ là thành phần triển khai, cụ thể hóa các lớp trừu tượng để tạo ra các thành phần và tập hợp các thành phần đó với nhau.



Director

Lớp này chịu trách nhiệm chính về việc điều khiển tất cả các thuật toán để tạo thành sản phẩm cuối cùng.





GIỚI THIỆU UML



Product

Lớp này sẽ định nghĩa loại của đối tượng phức tạp chúng ta chuẩn bị hướng đến.



Builder

Đây sẽ là lớp trừu tượng định nghĩa cho toàn bộ hành vi của quá trình xây dựng nên đối tượng phức tạp mà chúng ta nhắm đến.



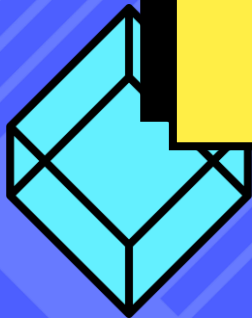
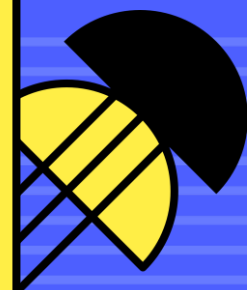
Concrete Builder

Đây sẽ là thành phần triển khai, cụ thể hóa các lớp trừu tượng để tạo ra các thành phần và tập hợp các thành phần đó với nhau.



Director

Lớp này chịu trách nhiệm chính về việc điều khiển tất cả các thuật toán để tạo thành sản phẩm cuối cùng.



Cái hay của Builder Design Pattern.

Đầu tiên:

- Đó là chúng ta không cần gọi tất cả các bước trong một lúc khi tạo đối tượng, nó hoàn toàn phụ thuộc vào nhu cầu của người dùng.

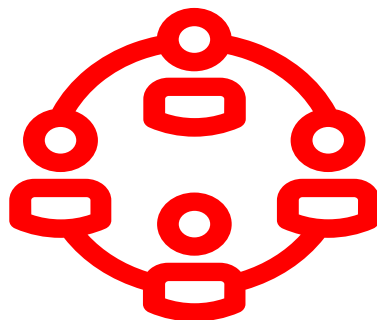
Cái hay của Builder Design Pattern.

Đầu tiên:

Khi khởi tạo các thuộc tính nhỏ trong đối tượng lớn phức tạp thì những hành động này sẽ luôn luôn tách biệt nhau và chẳng hề tác động gì tới những thuộc tính đã và đang có trong đối tượng lớn.

Tuy vậy:

- Dù muốn xây dựng 1 đối tượng phức tạp với tất cả các thuộc tính trong duy nhất 1 lần gọi đối tượng thì Builder pattern cũng sẽ giải quyết được.



**Builder không hoạt
động độc lập.**

Mối liên hệ giữa Builder và các mẫu khác

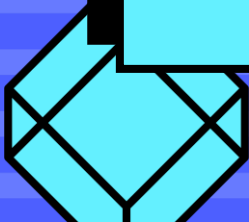


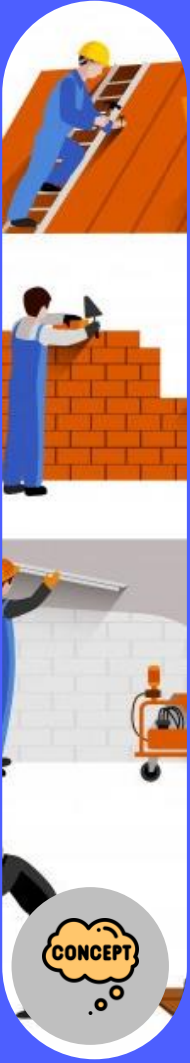
Kết hợp giữa Builder và Bridge.

Lúc này thì Lớp Director sẽ chịu trách nhiệm như là 1 lớp trừu tượng, còn những lớp Builder khác thì thực hiện công việc như thường.

Singletons.

Mẫu Builder có thể được triển khai như mẫu Singletons.







-ƯU ĐIỂM VÀ NHƯỢC ĐIỂM CỦA BUILDER DESIGN PATTERN.
Khi cân nhắc có nên sử dụng Builder Pattern hay không thì chúng ta sẽ điếm qua ưu nhược của chúng.




**Quản lí code
dễ dàng**

**Code có tính
đóng gói**

**Đối tượng
con luôn
khởi tạo
tách biệt**

**Thay đổi
tổng thể
dễ dàng**





Tạo nhiều code hơn bình thường

Giải pháp cho vấn đề lập trình thông thường


Chỉ bắt lợi khi bị lạm dụng



Tạo nhiều code hơn bình thường

Giải pháp cho vấn đề lập trình thông thường

Chỉ bất lợi khi bị lạm dụng




Tạo nhiều code hơn bình thường



Giải pháp cho vấn đề lập trình thông thường

Chỉ bắt lợi khi bị lạm dụng

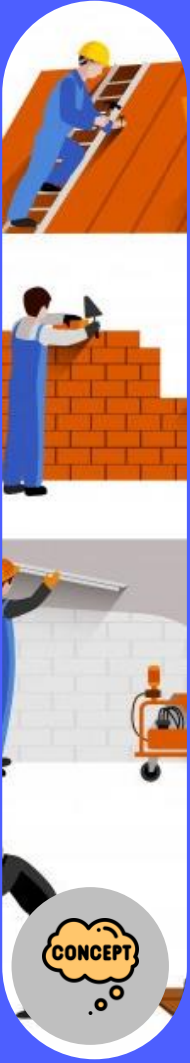


Tạo nhiều code hơn bình thường

Giải pháp cho vấn đề lập trình thông thường

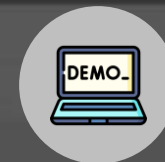
ABUSE

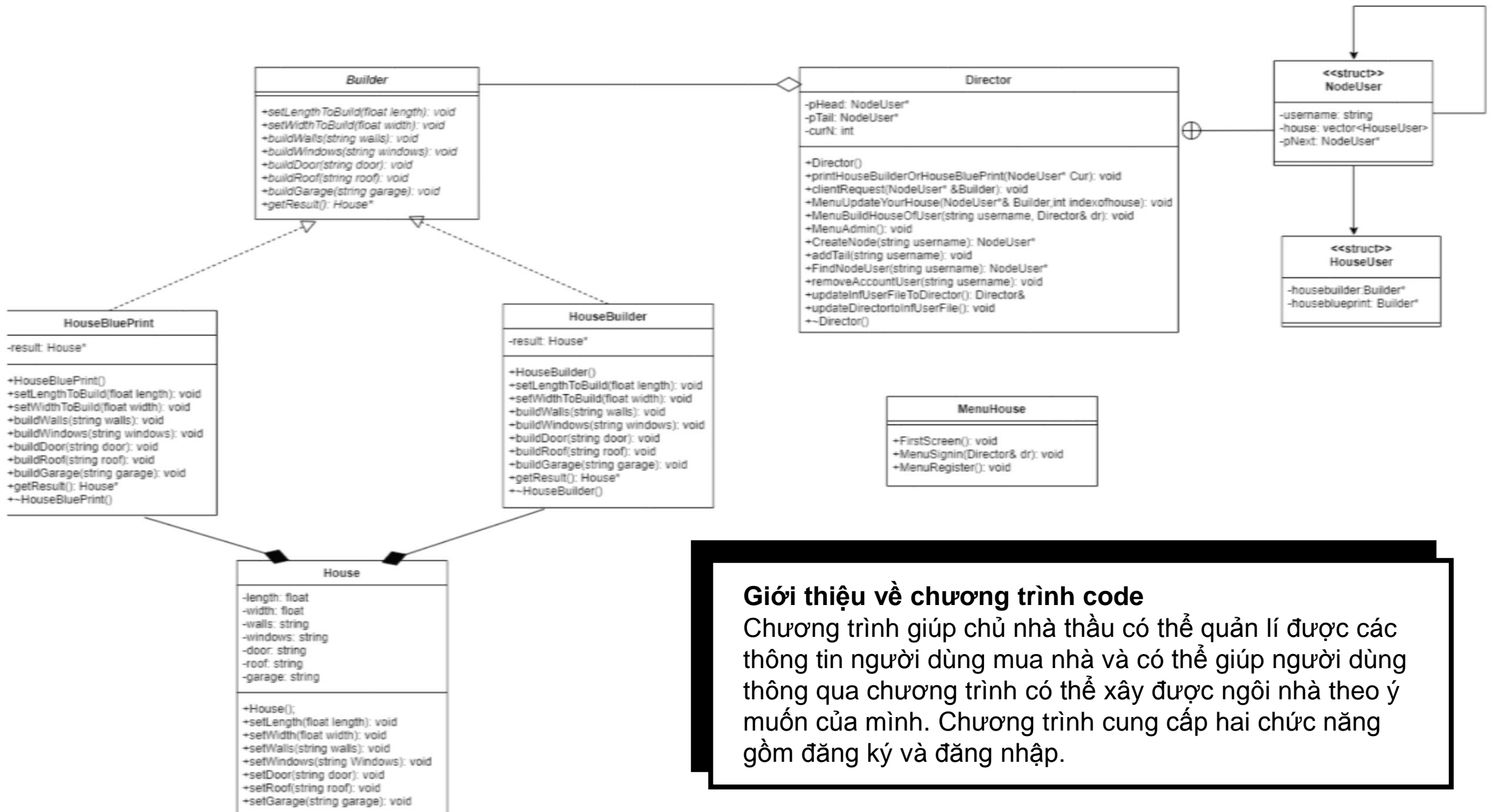
Chỉ bắt lợi khi bị lạm dụng





Sau đây sẽ là phần demo
UML và Code.





Giới thiệu về chương trình code

Chương trình giúp chủ nhà thầu có thể quản lí được các thông tin người dùng mua nhà và có thể giúp người dùng thông qua chương trình có thể xây được ngôi nhà theo ý muốn của mình. Chương trình cung cấp hai chức năng gồm đăng ký và đăng nhập.



Thanks !