-CustomHook: Muốn state thay đổi ở component nào thì dùng hàm thay đổi state ở component đó.

-Thread trong spring boot.

\*@Schedual:

+)fixedDelay = 1000 -> thời gian cố định thực hiện 1 tác vụ, luôn chờ tác vụ trước hoàn thành để thực hiện tác vụ tiếp theo.

+)fixedRate = 1000 -> thời gian cố định thực hiện 1 tác vụ, nó không quan tâm đến tác vụ trước đó có hoàn thành hay không để thực hiện tác vụ tiếp theo -> nghĩa là nó sẽ không trờ đợi tác vụ trước đó hoàn thành để thực hiện tác vụ tiếp theo, nếu tác vụ trước đó tốn 5 giây để hoàn thành trong khi fixedRate là 1 giây thì nó sẽ thực hiện các tác vụ trồng lên nhau.

+)@Asyn thêm vào cùng với @Schedual để thực hiện các tác vụ song song -> không đồng bộ.

+)initialDelay = 1000 => là hàm đó lần đầu tiên chạy tác vụ sẽ thực hiện thời gian chạy là 1000. Ví dụ ta có 1 hàm sử dụng fixedDelay = 2000 và initialDelay = 1000 thì lần đâu tiên nó sẽ sử dụng độ trễ của initialDelay và từ lần thực hiện công việc tiếp theo thì nó sẽ sử dụng độ trễ của fixedDelay.

+)cron: là thực hiện tính toán giờ để thực hiện 1 công việc, ví dụ như chạy 1 công việc vào mỗi 12h trưa.Cron sẽ sử dụng giờ địa phương của máy chủ cho biểu thức tính toán của nó.Ngoài ra nó còn cung cấp 1 tham số là zone để chỉnh múi giờ theo khu vực hoặc có thể chỉnh ở trong application properties.

+)Có thể sử dụng fixedDelayString hoặc fixedRateString để tham số hóa nó -> nghĩa là có thể sử dụng tham số được config ở trong application properties.

VD: @Scheduled(fixedDelayString = "${fixedDelay.in.milliseconds}")

@Scheduled(fixedRateString = "${fixedRate.in.milliseconds}")

@Scheduled(cron = "${cron.expression}")

+)Thông thường để mà thay đổi giá trị của fixedDelay hoặc fixedRate khi đó đang chạy là không thể -> Để thực hiện việc này thì nó cung cấp 1 interface là SchedulingConfigurer để thực hiện lập kế hoạch delay.

**\*Tích hợp thanh toán online:**

-