\* Quản lý hệ thống đặt phòng

+ QL cơ sở ks: Cho biết ks có những loại phòng nào, dịch vụ, giá cả, các tài khoản.

- Cập nhật tài khoản(thêm sửa xóa)

- Cập nhật loại phòng

- Cập nhật phòng

- Cập nhật dịch vụ

+ QL giao dịch khách hàng : tra cứu thông tin cần thiết, gửi yêu cầu đặt phòng,

gửi thông tin cá nhân và các yêu cầu về dịch vụ

- Tra cứu thông tin(tìm loại phòng trống)

- Đăng ký

- Đăng nhập

- Đặt phòng

- Yêu cầu phục vụ

+ QL kinh doanh ks: Giữ chỗ cho khách đặt phòng, thanh toán khi khách trả phòng

- Xử lý đơn đặt phòng

- Lập hóa đơn

- Cập nhật chi tiết hóa đơn

- In hóa đơn

- In báo cáo doanh thu định kỳ

- In báo cáo phòng trống

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Đặc tả xây dựng máy tìm kiếm**

Mô tả: Cho 1 họ các tài kiệu, được lưu trữ trên các trang web. Người sử dụng muốn tìm kiếm các tài liệu mà mình quan tâm. bằng cách nhập vào các từ khóa

\* Phân tích

- key: từ đơn, mở rộng từ khóa

- url chứa tài liệu có chứa mọi key

- Trong kết quả trả về 1 tài liệu xh trong 1 trang web

- Tên tài liệu là duy nhất

- Sắp xếp theo kết quả trả về theo thứ tự giảm dần của tổng số lần xuất hiện các từ khóa.

- Bổ sung từ khóa mới và cập nhật kết quả trả về

- Chọn 1 bản sánh, hiển thị tài liệu

\*\*\*\*\*\*\*\*\* **Date model**

\* key- tập các từ khóa

\* doc - họ các tài liệu

\* url - địa chỉ trang web

\* title - tập tên tài liệu

\* match - danh sách các bản sánh kết quả

\* number - số lần xuất hiện các cặp từ

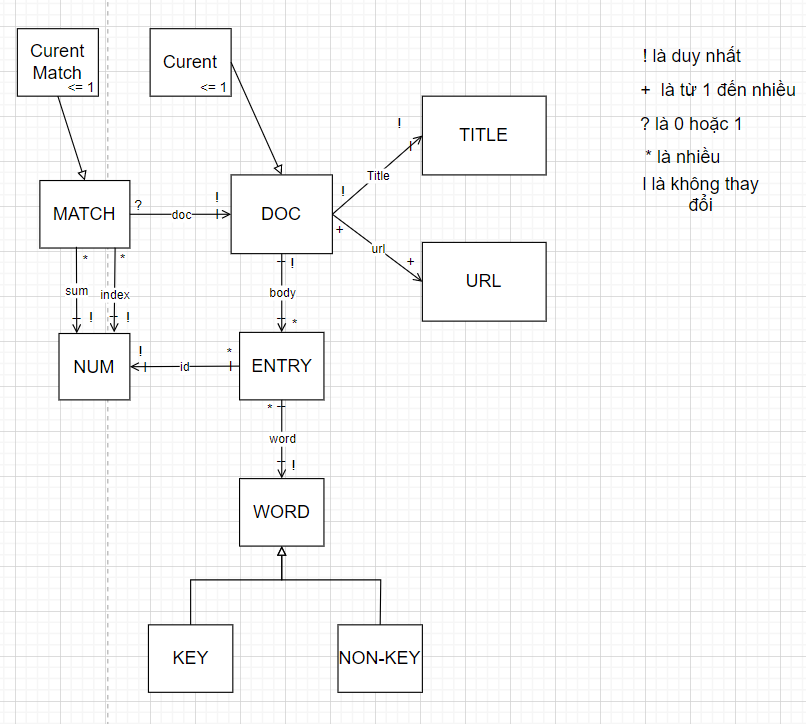
\* Entry - là cặp từ và vị trí của nó trong văn bản

\* word - tập các từ

\* nonKey(không phải là từ khóa) - cố định lưu ở trong file

\* currentMatch - bản sánh hiện thời

\* current - tài liệu hiện thời



* 1 match có sum là gì? => sumKey(doc, key) = là số lần xuất hiện của key trong doc
* m MATCH => m.sum = Tổng của sumKey(m.doc, k) trong đó k thuộc key
* Với mọi m1, m2 thuộc MATCH => m1.sum >= m2.sum -> m1.index <= m2.index
* Đặc tả các thao tác
* startMachine() – khởi động máy tìm kiếm nonKey
* query(String key) – truyền key để tìm kiếm
* queryMore(String r) – r không thuộc vào key => newKey = oldKey + r
* makeCurent(String t) – t thuộc title
* makeCurentMatch(int i) – chọn bản sánh thứ i là bản sánh hiện tại
* addDocuments(String u) – kiểm tra u thuộc url chưa. Nếu thuộc thì chỉ đọc bổ sung, ngược lại thì duyệt từ đầu.

**Search Engine**

**Class Engine{**

* Engnie() : Đưa các từ không phải từ khóa vào.
* Query QueryFirst(String key): truyền key để tìm kiếm => Trả về kết quả
* Query QueryMore(String word):
* Doc findDoc(String t): t là title của doc
* Query addDocs(String u):

**}**

* Chúng ta có 3 lớp: Engine, query, doc, titleTable, comm,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Báo cáo lần 3\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Tạo cơ sở dữ liệu(Có cả key, cả unique)
2. Chỉ rõ lệnh sql. Đặc tả 2 form dự kiến

---------------------------- Các mẫu

1. Mẫu adapter
   * Sử dụng khi đang sử dụng 1 đối tượng này muốn chuyển sang 1 đối tượng khác
   * Khởi tạo 1 đối tượng mới chứa đối tượng cũ
2. Mẫu facade
   * Khởi tạo 1 đối tượng trung gian, để tách 2 đối tượng
3. Mẫu composite
   * Sử dụng khi: Muốn phân cấp biểu diễn bộ phận – toàn bộ của các đối tượng
   * Thực hiện: Dùng chung phương thức operation
4. Mẫu singletion
   * Sử dụng khi chỉ muốn tạo duy nhất 1 đối tượng
   * Thực hiện: Khi tra null 2 lần trước khi tạo đối tượng. Lần thứ nhất để phục vụ đa luồng có thể sử dụng, lần thứ 2 để khẳng định mỗi luông chạy chỉ tạo duy nhất 1 dối tượng.
5. A
6. A
7. Mẫu observer:
   * Dùng khi:
     + Khi trừu tượng có hai khía cạnh, một phụ thuộc vào cái kia. Đóng gói các khía cạnh này trong các đối tượng tách bạch nhau cho phép bạn đa dạng và tái sử dụng chúng độc lập.
     + Khi một thay đổi của đối tượng này yêu cầu thay đổi các đối tượng khác và bạn không biết có bao nhiêu đối tượng cần phải được thay đổi.
     + Khi một đối tượng có thể thông báo cho các đối tượng khác mà không cần giả thiết về ai là các đối tượng này. Nói cách khác, bạn không muốn các đối tượng này quá gắn chặt nhau.
   * Thực hiện:
     + s
8. Mẫu mediator(Mẫu trung gian)
   * Dùng khi:
     + Một tập các đối tượng trao đổi theo các cách hoàn toàn xác định, nhưng phức tạp. Các sựphụ thuộc lẫn nhau là không có cấu trúc và khó hiểu.
     + Tái sử dụng một đối tượng là khó bởi vì nó tham chiếu đến và trao đổi với rất nhiều đối tượng khác.
     + Một hành vi mà phân tán giữa một số lớp cần phải được điều chỉnh không cần nhiều lớp con.
   * Thực hiện: Tạo 1 đối tượng liên kết trung.
9. Mẫu proxy(Mẫu ủy quyền - đại diện)
   * Dùng khi: muốn trì hoãn tạo 1 đối tượng. Trì hoãn tạo đối tượng vì tốn kém bộ nhớ và thời gian. Trì hoãn bằng cách tạo 1 đối tượng đại diện. Ở đây có một số tình huống chung ở đó mẫu proxy được áp dụng:
     + Remote Proxy cung cấp đại diện địa phương cho một đối tượng ở không gian địa chỉ khác
     + Virtual Proxy tạo các đối tượng đắt theo nhu cầu.
     + Protetion proxy kiểm soát truy cập đến đối tượng gốc. Protection proxy là hữu ích khi các đối tượng có các quyền truy cập khác nhau. Muốn phân quyền thì đưa ông đại diện bảo vệ vào.
10. Mẫu CHAIN OF RESPONSIBILITY(Dãy các trách nghiệm)
    * Mẫu này là tất cả về kết nối các đối tượng trong danh sách móc nối, một thông báo sẽ đi dọc theo danh sách, nó được xử lý bởi đối tượng đầu tiên mà được thiết lập để làm việc với loại
    * thông báo này
    * Dùng khi: Có các vấn đề cần giải quyết thì xem giải quyết được cái nào thì giải quyết, còn đâu chuyển cho đối tượng khác.
    * Thực hiện:
11. Mẫu FLYWEIGHT(Mẫu hạng ruồi)
    * Dùng khi:
    * Thực hiện:
12. Mẫu BUILDER(Mẫu xây dựng)
    * Dùng khi: Một đối tượng có quá nhiều thuộc tính khác nhau.

Sử dụng mẫu Builder khi

 Thuật toán để tạo đối tượng phức tạp cần là độc lập khỏi các phần mà tạo nên đối tượng và

chúng được lắp ghép như thế nào

 Quá trình tạo cần cho phép các thể hiện khác nhau cho các đối tượng mà được tạo.

* + Thực hiện: Ko tạo hàm tạo mà tạo ở bên ngoài rồi gán lại đối tượng bằng hàm set.

1. Mẫu FACTORY METHOD(Mẫu phương thức của nhà máy)
   * Dùng khi: Sử dụng mẫu Factory Method khi

 Một lớp không biết trước được lớp của các đối tượng mà cần được tạo.

 Một lớp muốn các lớp con của nó đặc tả các đối tượng mà nó tạo.

 Các lớp ủy quyền tương ứng cho một trong số các lớp con bổ trợ và bạn muốn cục bộ hóa hiểu biết này của lớp con bổ trợ mà được ủy quyền

1. Mẫu ABS FACTORY METHOD(Mẫu phương thức của nhà máy)

* Khi 1 hệ thống cần độc lập với việc các sản phầm cần tạo.

1. Mẫu thiết kế Prototype(Dùng deep-clone)

* Dùng khi muốn tạo đối tượng mới nhưng không muốn tạo qua hàm tạo thì ta nhân bản đối tượng đã có để sử dụng.
* Quyền truy cập là 1 map có key là 1 module(vai trò) value là thao tác, thông tin được sử dụng.
* Sử dụng mẫu thiết kế Prototype khi một hệ thống cần là độc lập với việc các sản phẩm của nó
* được tạo, kết hợp và thể hiện như thế nào và
*  Khi các lớp để khởi tạo được đặc tả trong thời gian thực, chẳng hạn, bằng tải động hoặc
*  Để tránh việc xây dựng một phân cấp lớp các factory mà chạy song song với phân cấp lớp
* của sản phẩm.
*  Khi các khởi tạo của một lớp có thể có một trong chỉ một số ít kết hợp khác nhau của trạng
* thái. Nó có thể là thuận tiện hơn để cài đặt một số tương ứng các nguyên mẫu và nhân bản
* clone chúng hơn là khởi tạo lớp bằng tay, mỗi lần với một trạng thái phù hợp.

1. Mẫu thiết kế memento

* Dùng khi: Bản chụp nhanh trạng thái (hoặc một phần) của đối tượng cần được lưu sao cho nó có thể được khôi phục về trạng thái này về sau.
* Dùng gilog để lưu trữ trạng thái từ lúc savePoint đến lúc chương trình lỗi.
* CareTaker được sử dụng để lưu và truy vấn đối tượng memento mà biểu diễn trạng thái của đối tượng originator.
* Dùng cấu trúc HashMap<String, Memento>

1. Mẫu thiết kế template(Bộ khung)

* Dùng khi: Thiết kế 1 thuật toán bao gồm 1 số phương thức dùng chung và 1 số phương thức phải tự cài đặt lại.

1. Mẫu thiết kế State(Trạng thái)

* Dùng khi: 1 hành vi của đối tượng phụ thuộc vào trạng thái của nó

1. Mẫu thiết kế Strategy(Mẫu chiến lực): Tách các chiến lược ra 1 interface khác. Giúp cho code linh hoạt

* Dùng khi: Có nhiều đối tượng chỉ khác nhau ở hành vi của chúng

1. Mẫu thiết kế command(Yêu cầu):

* Dùng khi:

1. Mẫu thiết kế interpreter

* Dùng khi: