Bài Báo Cáo

**1. Dependency injection (DI) và những thứ liên quan đến nó**

Tight coupling hay “liên kết rằng buộc ” là một khái niệm trong java ám chỉ việc mối quan hệ giữa các Class quá chặt chẽ . Khi yêu cầu thay đổi logic hay một class bị lỗi sẽ dẫn tới ảnh hưởng tới toàn bộ các class có liên quan đến class đang thay đổi

* Loosely-coupled là cách ám chỉ việc làm giảm phụ thuộc giữa các Class với nhau
* Vậy câu hỏi là có kỹ thuật nào khắc phục nhược điểm của Tight coupling và áp dụng khái niệm Lossely-coupled không ? từ đó ta sinh ra thêm một khái niệm đó là Dependency injection (DI) là một kỹ thuật lập trình giúp lập trình viên có thể dễ dàng bảo trị code của mình , linh hoạt , dễ chỉnh sửa code nói chi tiết hơn đó là “ Dependency Injection là việc các Object nên phụ thuộc vào các Abstract Class và thể hiện chi tiết của nó sẽ được Inject vào đối tương lúc runtime”
* Ta có các cách để Injection vào đối tượng như là
  + - **Constructor Injection**
    - **Setter Injection**
    - **Interface Injection**
* Tổng hợp các Anotation hay sử dụng trong Sping boot là :
  + @SpringBootApplication chúng ta dùng Anotation này để chi cho thằng spring biết rằng đây là hàm main , khi chạy project nó sẽ chạy bắt đầu từ hàm được gắn trên một class
  + @Component là một Annotation đánh dấu trên các Class để giúp Spring biết nó là một Bean
  + @Autowired tự động tiêm một instance từ trong container vào đối tượng

## 2. Chạy nhiều môi trường với spring Profile

## - Spring Profile là một core feature trong Spring Framework , cho phép chúng ta cấu hình ứng dụng , active Bean tùy theo môi trường . Sẻ có một ví dụ như thế này “================================”

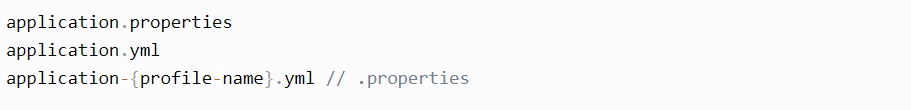
## Sớm nhận ra những khó khăn trong việc config khi xây dựng ứng dụng , nên spring đã cho ra đời Spring Profiles để giải quyết điều này

## Vậy nó như thế nào , vào sử dụng nó ra sao , vậy chúng ta đi tiếp nhé

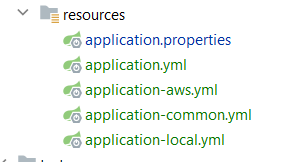
2.1 Tạo file config

Spring Profiles có sẵn trong Frameword rồi nên bạn không cần thêm bất kì thư viện nào khác .

Để sử dụng , các bạn tạo file Config tại thư mục resources trong project . Mặc định Spring sẽ nhận các file có tên như sau



Ví dụ như mình có 2 môi trường là “Local” và aws,thì mình sẽ tạo ra file như thế này:



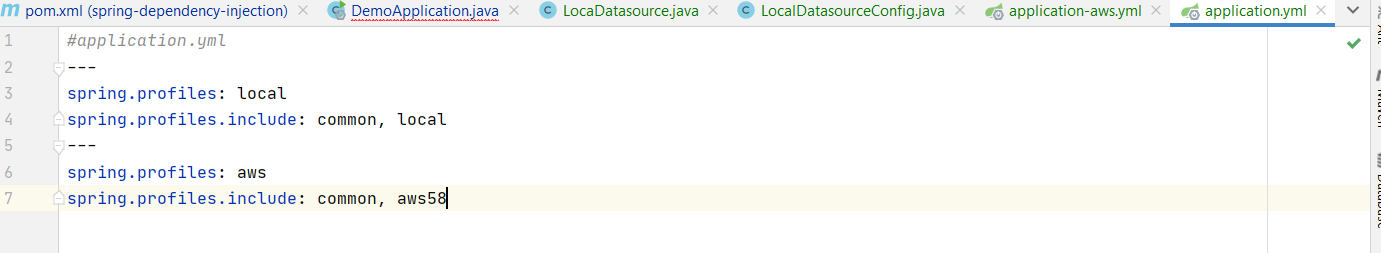
application.properties là file config chính khai báo các Enviroment.

application-local.yml là file chỉ sử dụng khi chạy chương trình ở local

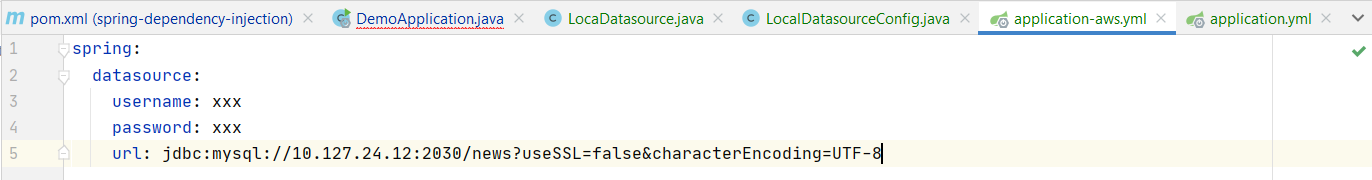
application-aws.yml chỉ là sử dụng khi chạy aws

application-common.yml là những config dùng chung , môi trường nào cũng cần .

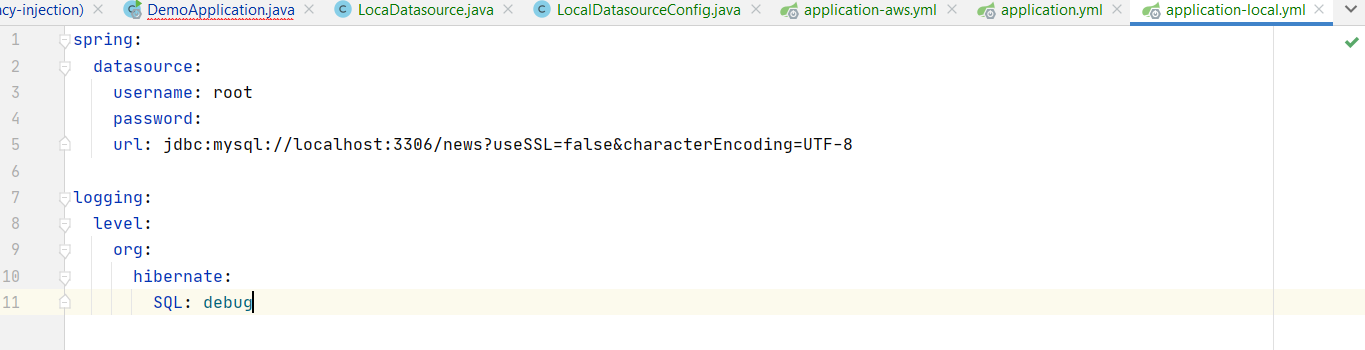
Bây giờ chúng ta khai báo từng file như sau :

Application.yml:

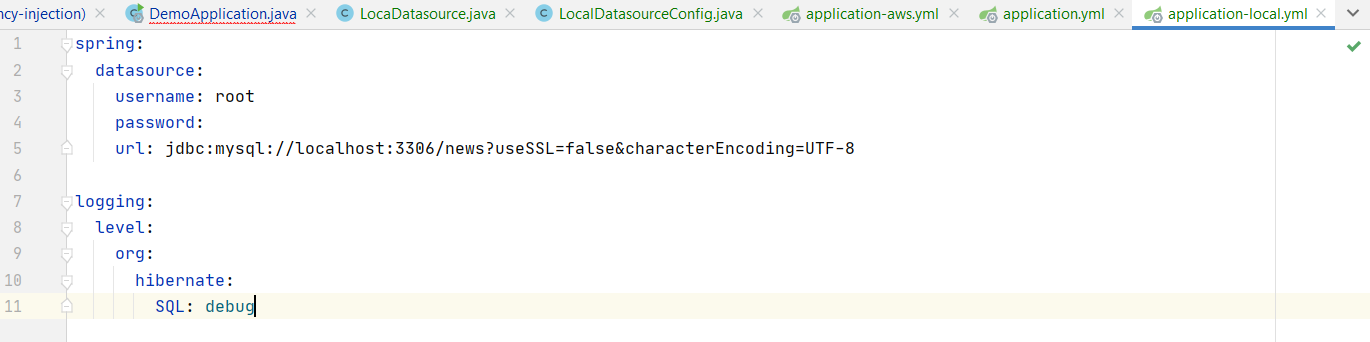
application-aws.yml :



application-local.yml :



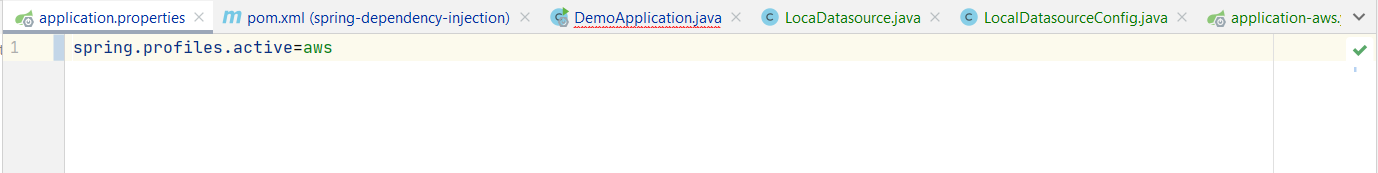
application-common.yml



Tadaa, xong! chúng để ý trong file application.yml mình có khai báo 2 môi trường là local và aws. Tại mỗi môi trường sẽ include (bao gồm) các file config như kia. Khi mình kích hoạt aws chẳng hạn, Spring sẽ load tất cả config có trong application-common.yml và application-aws.yml.

2.2.Kích hoạt config

Để sử dụng Profiles chúng ta làm như sau :

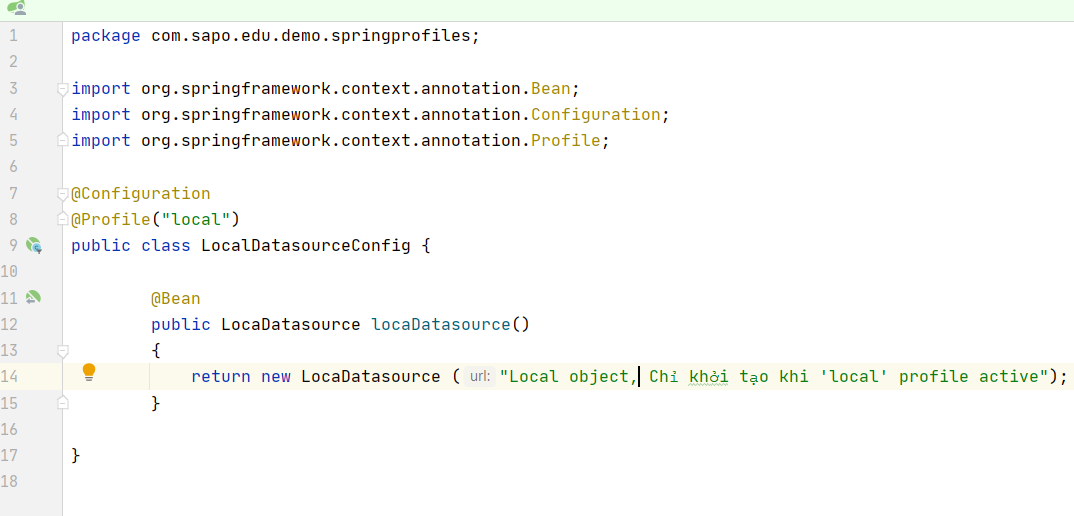
Sử dụng spring.frofiles.active = aws trong application.properties 

Chúng ta sẽ tạo ra một class là LocalDatasource , và kiểm tra xem , khi thay đổi profile thì nó sẽ được tạo ra hay không nhé

LocalDatasource.Java



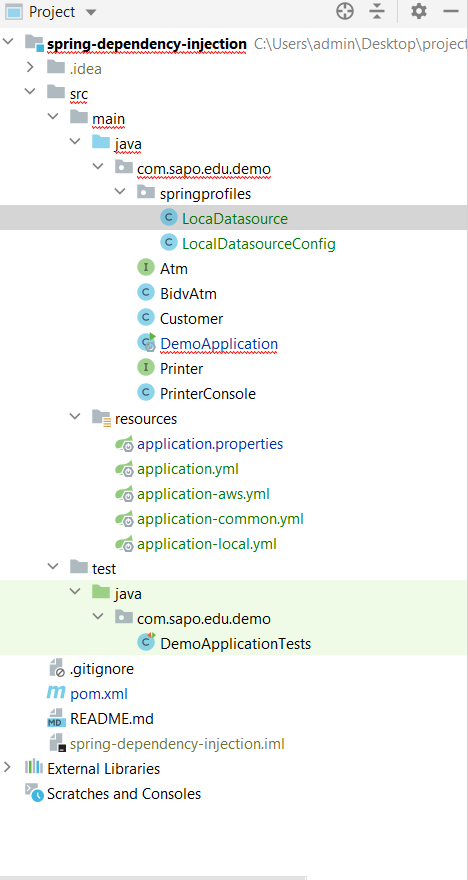
LocalDatasourceConfig



Đây là trương trình test



3.File Cấu Hình application.properties



Application.properties nằm trong thư mục resources nó dùng để cấu hình các file liên quan đến port , cơ sở dữ liệu , … tất cả các thứ liên quan đến cấu hình trong project spring boot của chúng ta