# Mục tiêu tài liệu

* Có kiến thức cơ sở Spring Framework
* Project Spring MVC
* Project Spring Rest API

Tài liệu này dựa vào các công nghệ thời điểm hiện tại đươc sử dụng (2021).

|  |  |
| --- | --- |
| CSDL | MySQL (Dùng XAMPP) |
| Spring version | 5.xx |
| Spring Boot | 2.4.4 |
| Java version | 11 (Recommend) |
| Git | Github |
| IDE | Eclipse (có thể dung Spring Tool Suite) |
| Docker | Nếu có thể |

# Yêu cầu tài liệu

## Kiến thức

* Java core
* Mô hình MVC, DAO design pattern, Mô hình 3 lớp
* Spring Bean
* Có kiến thức hoạt động của JWT
* Có kiến thức cơ bản Web Service, API, Soap, Rest, RestFul service
* JPA

## Môi trường

* Java 11
* Eclipse hoặc Intelli J

## Các công nghệ liên quan

* Spring Security
* Spring MVC
* Spring Bean
* JWT: JSON Web Tokens

# Cơ Sở Dữ Liệu

## Kiến thức vào mẹo

* Mẹo: Trong Spring JPA nó sẽ hộ trợ tao DB với class java luôn. Và có thể import cấu trúc DB có sẵn với tên file chính xác (schema.sql) và data có sẵn bằng lệnh REPLACE INTO (kiểu như INSERT INTO) bằng file chính xác tên (data.sql).

Chính nhờ vậy có thể tạo các class Java và không cần tạo SQL. Nhưng chính vì thể cần viết rõ thuộc tính trong @Column() độ dài, unique không, nullable không

Link tham khảo mẹo:

<https://www.baeldung.com/spring-boot-data-sql-and-schema-sql>

* Việc tạo các class dung chung các cột. Có thể tạo ra 1 class chứa các cột cơ bản chung Ví dụ: (ngày tạo, người tạo, người update, ngày update) có ở demo dưới.

Việc như vậy có thể tạo supper class (class cha, các entity khác kế thừa lại) sau đó sử dụng kế thừa. Nhưng việc này có 2 cách sử dụng @MappedSuperclass và @Entiy

Hai cách này cơ chế hoạt động khác nhau và khuyên dung nên là: @MappedSuperclass. Nếu dùng @Entity thì JPA hiểu là class supper là 1 bảng, @MappedSuperclass thì là không. Còn cụ thể detail ở link dưới

Link tham khảo <https://docs.oracle.com/javaee/6/tutorial/doc/bnbqn.html>

## Các chức năng cơ bản

* Đăng ký đăng nhập
* CRUD các bảng
* Mua bán đặt hàng
* --------

## Thiết kế CSDL cơ bản

Các bảng tìm hiểu sẽ làm chức các chức năng. Việc thiết kế CSDL ở nên xây dựng từ việc nhìn nhận bài toán thực tế và phân chia các tập đối tượng, tập thực thể (Table) và xác định replation giữa các tập thực thể với nhau dựa vào logic nghiệp vụ. Việc không nên nhất là dựa vào thiết kế Giao diện để làm ra các bảng (Thói quen xấu và sai khá nhiều). Các bảng phải đáp ứng thừa hoặc ít nhất bằng số lượng trên thiết kế giao diện.

Sau đây là 1 số bảng để đáp ứng được bài toán thử nghiệm để học công nghệ chứ không đi vào sâu nghiệp vụ (Đáp ứng được 1-1, m-m, 1-n)

Trong một số nghiệp vụ chưa chắc chắn sự liên kết thì việc kết nối m-m là việc tốt nhất, vì vẫn đảm bảo sự mở rộng, tất nhiên là việc xác định được DB trước khi viết code là điều ưu tiên nhất.

### Base Entity



Base: Các bảng sẽ đều có các cột

|  |  |
| --- | --- |
| createTime | Thời gian tạo |
| createUser | Người tạo |
| updateTime | Thời gian update |
| updateUser | Người update |

Các bảng cho đăng nhập

### User

|  |  |
| --- | --- |
| id |  |
| username |  |
| password |  |
| firstName |  |
| lastName |  |
| enable |  |

### Role

|  |  |
| --- | --- |
| id |  |
| roleName |  |

### User\_Role

|  |  |
| --- | --- |
| roleId |  |
| userId |  |
| other | String |

Các bảng cho logic mua bán

### Product

|  |  |
| --- | --- |
| id |  |
| name |  |
| price |  |
| slug | Url được SEO |
| imageUrl1 |  |
| imageUrl2 |  |
| imageUrl3 |  |
| imageUrl4 |  |
| categoryId |  |
|  |  |

### ProductReview

|  |  |
| --- | --- |
| id |  |
| productId |  |
| star | Số sao từng người đánh giá |
| comment | Comment tường người đánh giá. Nếu có thể tách thành bảng comment |

### Order

|  |  |
| --- | --- |
| id |  |
| status | Đã commit mua hay chưa, Nếu chưa thì là giỏ hàng |

Product\_Order

|  |  |
| --- | --- |
| productId |  |
| orderId |  |
| quantity |  |
| other |  |

### Category

|  |  |
| --- | --- |
| id |  |
| name |  |
| description |  |
| imageUrl |  |
| parentId |  |

### Cart

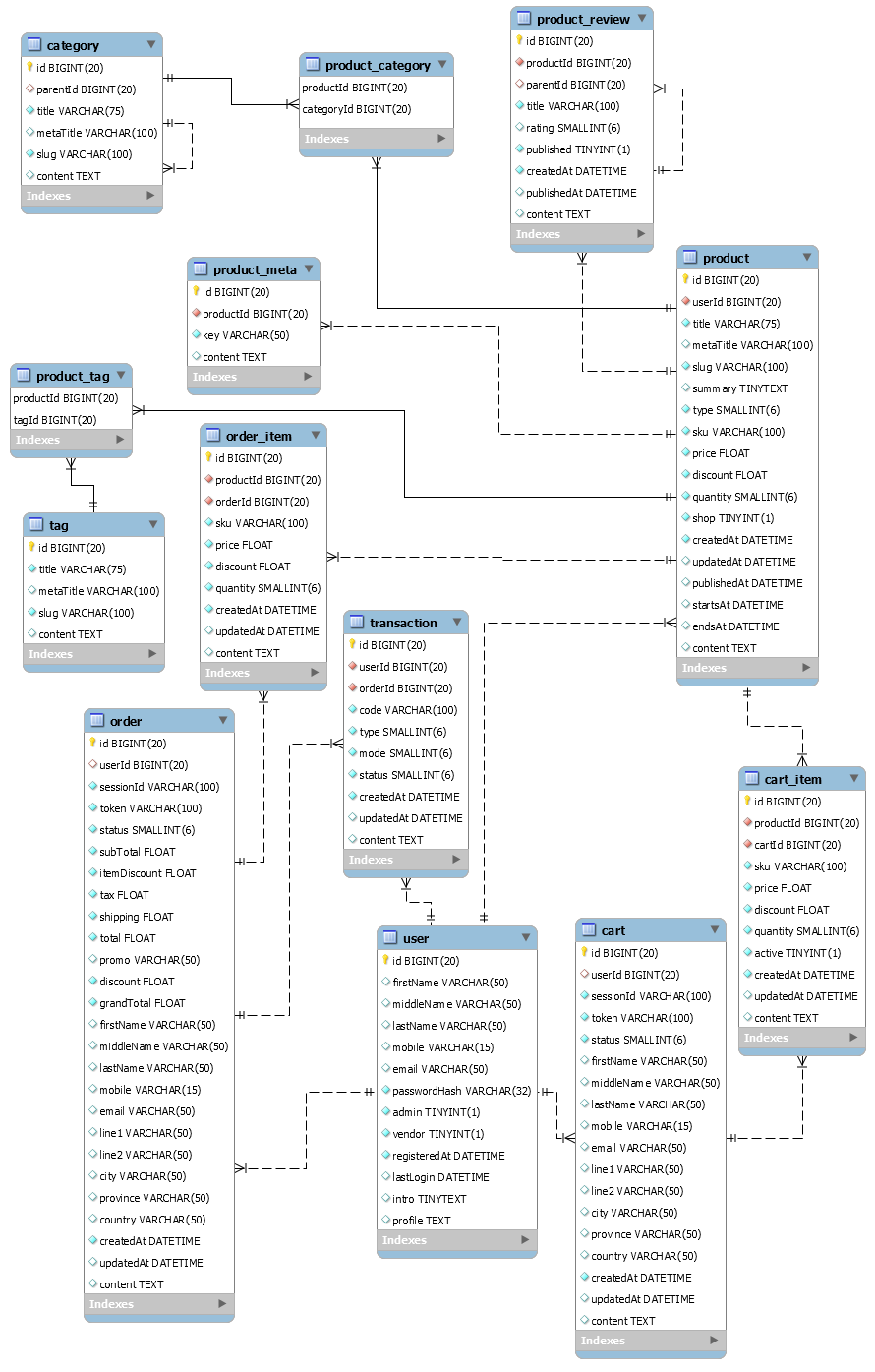
|  |  |
| --- | --- |
| id |  |
| name |  |
| description |  |
| status | Đã thanh toán, chưa thanh toán,… |
| userId | Id người đăng nhập |
| amount |  |

### CartItem

|  |  |
| --- | --- |
| id |  |
| cartId |  |
| productId | Mã sản phẩm |
| quantity | Số lượng sản phẩm |
| name |  |
| description |  |
| price | Gía của thời điểm mua |
| discount | % giảm giá thời điểm mua |

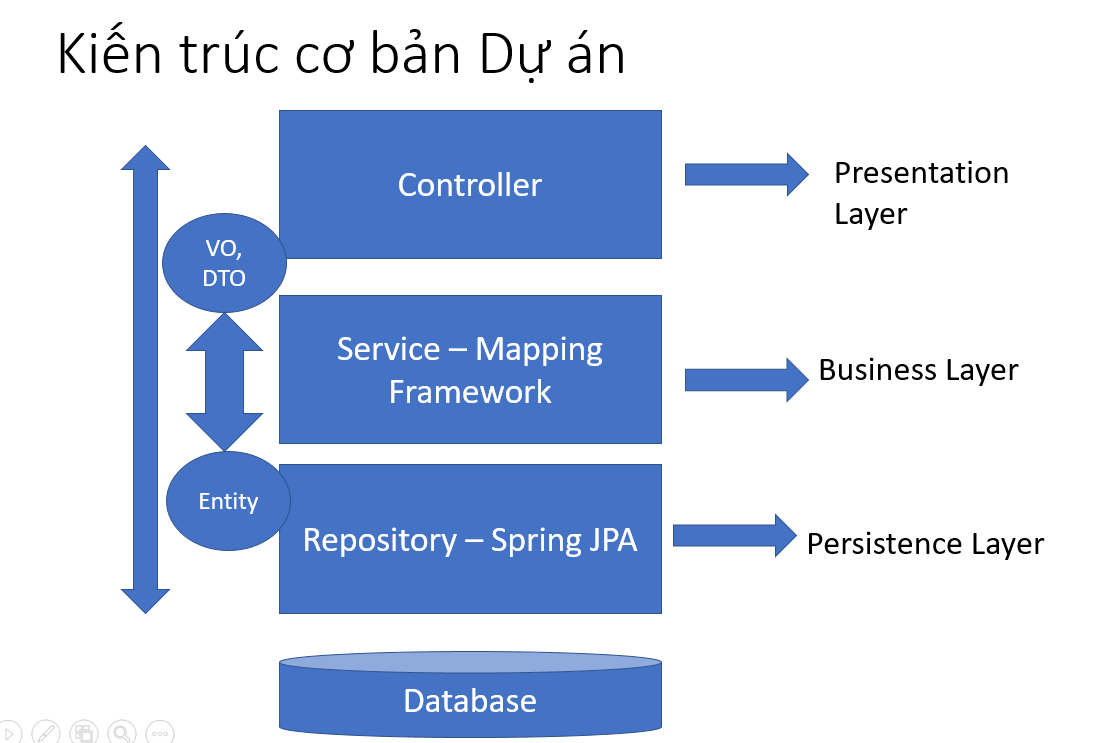
Thiết kế CSDL có thể dựa vào link sau:

<https://mysql.tutorials24x7.com/blog/guide-to-design-database-for-shopping-cart-in-mysql>



Hình 1: Mô tả các bảng trong CSDL (mẫu)

# Sơ lược Mô hình 3 lớp



Hình 1: Mô hình 3 lớp được mô tả trên hình

Mô hình 3 lớp nên tuân thủ quy tắc tầng nào làm nhiệm vụ tầng ấy (Single responsibility ciple - SOLID).

Cách code mô hình 3 lớp kết hợp DAO pattern đó là mỗi tầng sẽ tạo ra interface và implementation để có thể sử dụng.

Phân chia công việc các tầng mô hình 3 lớp

## Presention Layer

* Thành phần là Controller và view
* Nhiệm vụ chủ yêu là nhận request và trả về response cho người dung
* Chức năng xử lý chủ yếu là Validation

## Business Layer

* Thành phần chủ yếu là các service xử lý nghiệp vụ
* Nhiệm vụ chủ yêu xử lý nghiệp vụ và biến đổi dữ liệu (Đẩy thành Persistence Object (Entity) cho Persisten Layer xử lý) hoặc là Model cho Controller show lên View (MVC)
* Chức năng xử lý lý nghiệp vụ

## Persistence Layer

* Thành phần chủ yếu là các Repository
* Thực hiện vòng đời của các Persistence Object (merge, persit, delete,…)
* Đáp ứng đầy đủ các câu lệnh SQL (tối thiểu là CRUD)

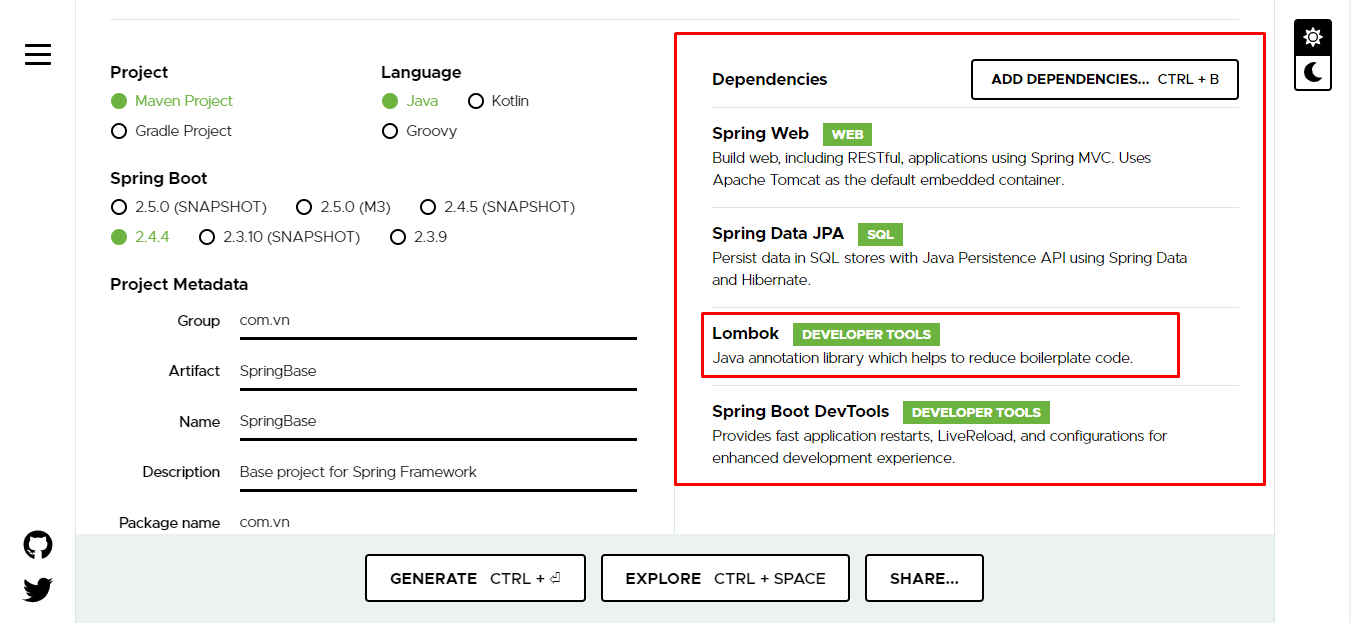
Sau khi hiểu rõ các thành phần thì chúng ta không nên viết đan xen các logic và luồng xử lý để trành việc không quản lý được code

# Tạo project Setting cơ bản

## Step1: Tạo project Spring

Vào trang: <https://start.spring.io/>

Thực hiện generate project với các thành phần module như hình



Hãy cài đặt Lombok cho IDE của mình.

## Step1.1 Thêm thư viện JDBC

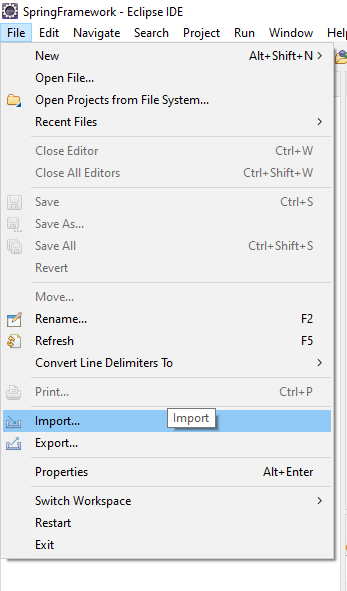
ở đây sẽ thêm thư viện cho MySQL.

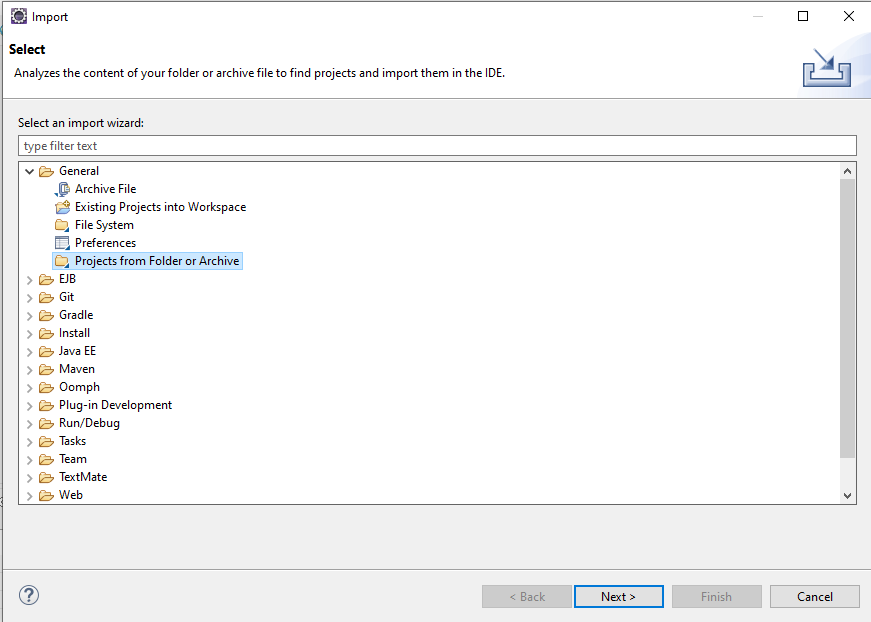
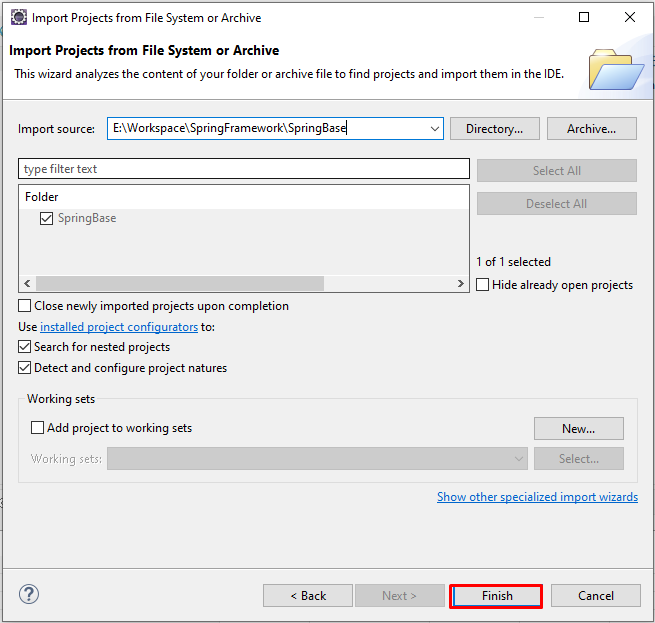
|  |
| --- |
| <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>8.0.21</version>  </dependency> |

Pom thời điểm hiện tại



## Step2: Giải nén project và import Eclipse



Chờ đợi Project import thành công thì thực hiện sử dụng Maven build

1. Chuột phải project > Run as > Maven Build
2. Chuột phải project > Maven > Update Project

Step3: Tạo các package cơ bản

Src/ main / java tạo các package có thể sử dụng trong thời gian tới

|  |  |
| --- | --- |
| Tên | Ý nghĩa |
| com.vn | Chứa Main của Spring Boot, Đóng vai trò làm base package |
| com.vn.config | Chứa các config |
| com.vn.controller | Chứa các Controller (MVC) |
| com.vn.entity | Chứa các Entity, Persistence Object |
| com.vn.repository | Chứa các repository |
| com.vn.service | Chứa Business Logic |
| com.vn.util | Chứa các Utility cho ứng dụng (đọc file,…) |
| com.vn.exception | Xử lý các Exception trong hệ thống |
| com.vn.vo | Chứa các Value Object |

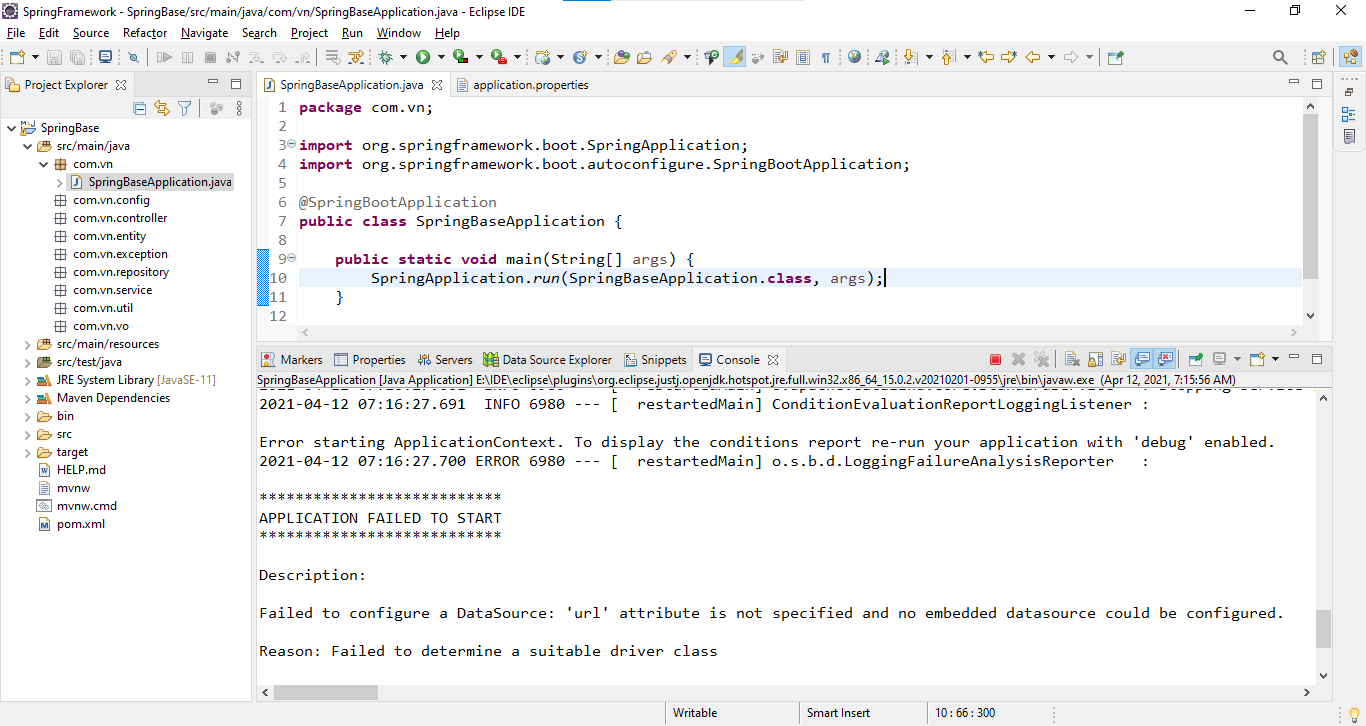
Bảng 1: Phân chia các packge cơ bản

Chú ý: Điểm mấu chốt ở đây là base packge từ file main của ứng dụng chạy, còn không thì phải sử dụng component scan để có thể biết vị trí các component cần quét của Spring.

Để tránh phức tạp thì lựa chọn packge gốc chứa packge component bên trong như bảng trên

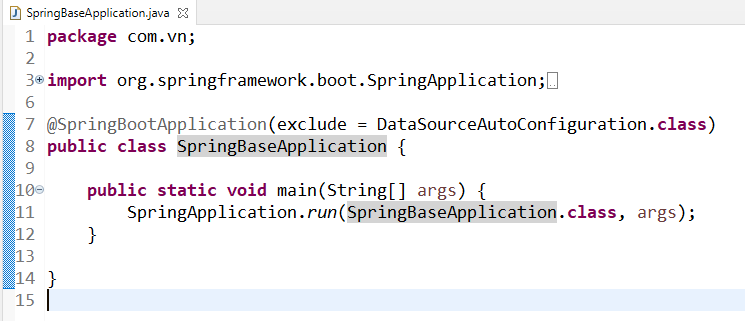
## Step3: Cấu hình Spring JPA application.properties

Khi chạy ứng dụng sẽ bị lỗi



Nguyên nhân lỗi sẽ là Spring Boot application sẽ call tới các cấu hình tự động để xử lý được

1. Cấu hình luôn cho Spring JPA để kết nối CSDL luôn
2. Exclude cấu hình data source tự động như hình



Tuy vậy chúng ta sẽ không sử dụng code như trường hợp này

Cấu hình cho Spring JPA

|  |
| --- |
| # data source  spring.datasource.url=jdbc:mysql://localhost:3306/ECommerceApplication?useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC  spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect  spring.datasource.username=root  spring.datasource.password=  spring.jpa.properties.hibernate.show\_sql=true  #create, update, create-update, create-drop, none  spring.jpa.hibernate.ddl-auto=update  spring.datasource.initialization-mode=always  spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl |

Giải thích:

* spring.jpa.hibernate.ddl-auto= create-drop, create sử dụng khi lần đầu chạy ứng dụng nó sẽ tạo tự động các bảng. Khi ấy nó sẽ import file tên là “data.sql” nếu có và sử dụng file “schema.sql” nếu có.
* spring.jpa.properties.hibernate.dialect ở đây lựa chọn MySQL5InnoDBDialect thì sẽ không tạo được kết nối DB bởi các Entity trong code java mà nên sử dụng MariaDBDialect nếu có.
* spring.jpa.hibernate.naming.physical-strategy sử dụng PhysicalNamingStrategyStandardImpl tức là cột sẽ bằng variable name trong Entity

## Step4: Tạo Repository Layer

ở đây sẽ tạo các Repository, Custom Repository và các implement các repository nếu có.

### UserRepository

### RoleRepository

### UserRoleRepository

### ProductRepository

### ProductReviewRepository

### OrderRepository

### CategoryRepository

## Step5: Tạo các service interface

## Step6: Tạo các service implementation

# Tạo Spring API project

Sau khi đã đạt được Spring Base project ở các step trên thì tiếp theo tìm hiểu về Spring REST API

Thời điểm hiện tại của tài liệu này, Spring có thể viết Spring REST APIs hoặc Spring Reactive REST APIs.

## Thế nào là chuẩn REST API