# Mục tiêu tài liệu

Có kiến thức cơ sở Spring Framework

Project Spring MVC

Project Spring Rest API

# 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

## 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

## 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 ch các chức năng

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 |  |
| imageUrl1 |  |
| imageUrl2 |  |
| imageUrl3 |  |
| imageUrl4 |  |
| categoryId |  |

Order

|  |  |
| --- | --- |
| id |  |
| status |  |

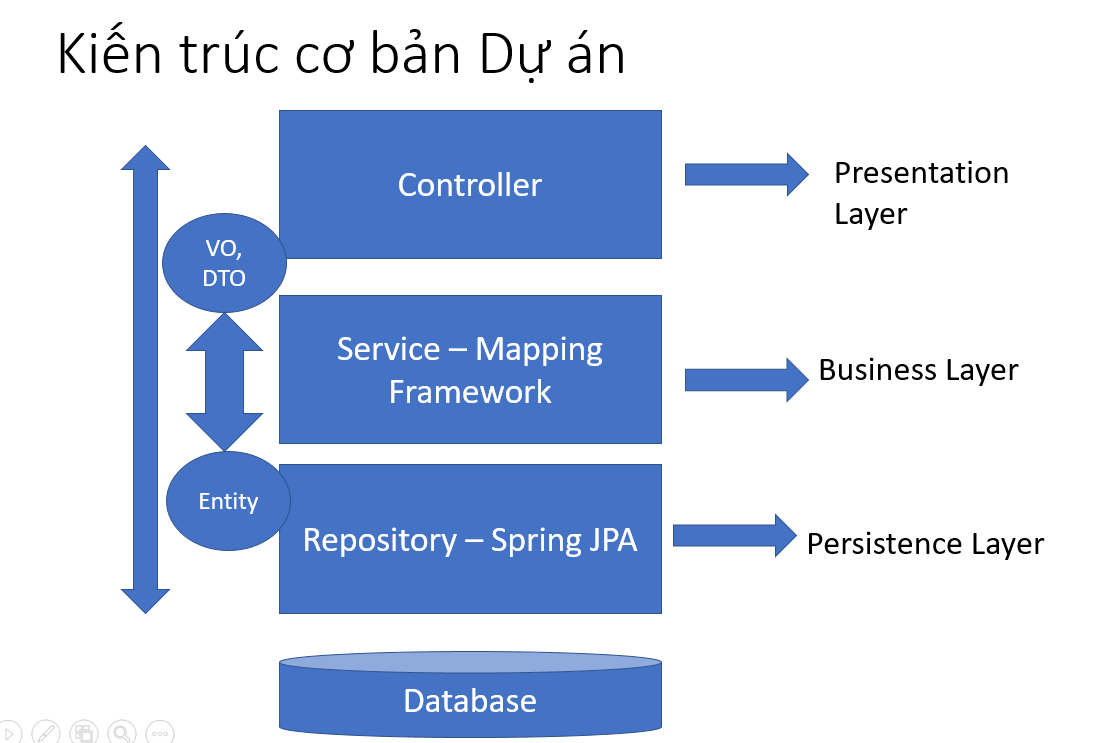
Product\_Order

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

Category

|  |  |
| --- | --- |
| id |  |
| name |  |
| description |  |
| ImageUrl |  |

# 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)

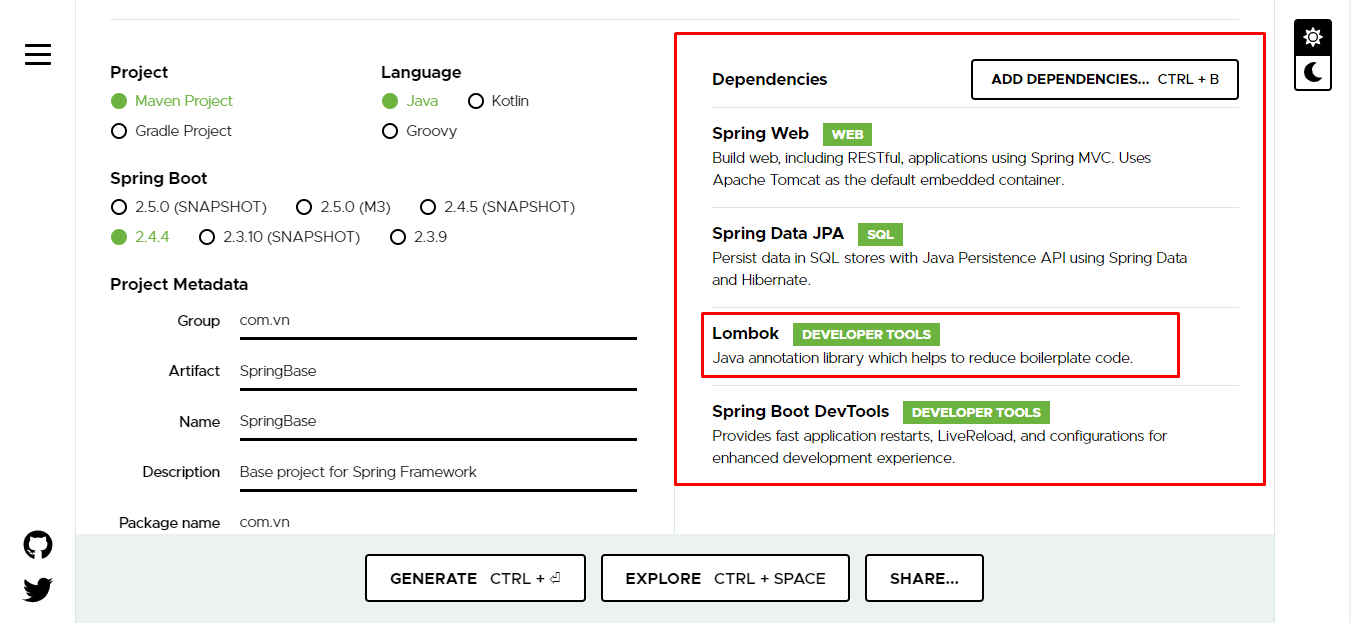
Sau khi hiểu rõ các thành phần thì chúng ta không nên viết đan xen các logic và luồng xử lý để trành việc không quản lý được code

# Tạo project Setting cơ bản

## Step1: Tạo project Spring

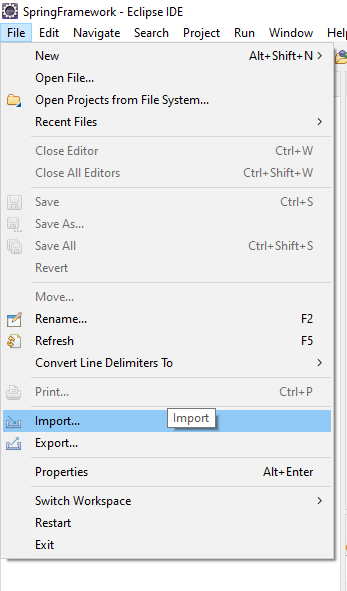
Vào trang: <https://start.spring.io/>

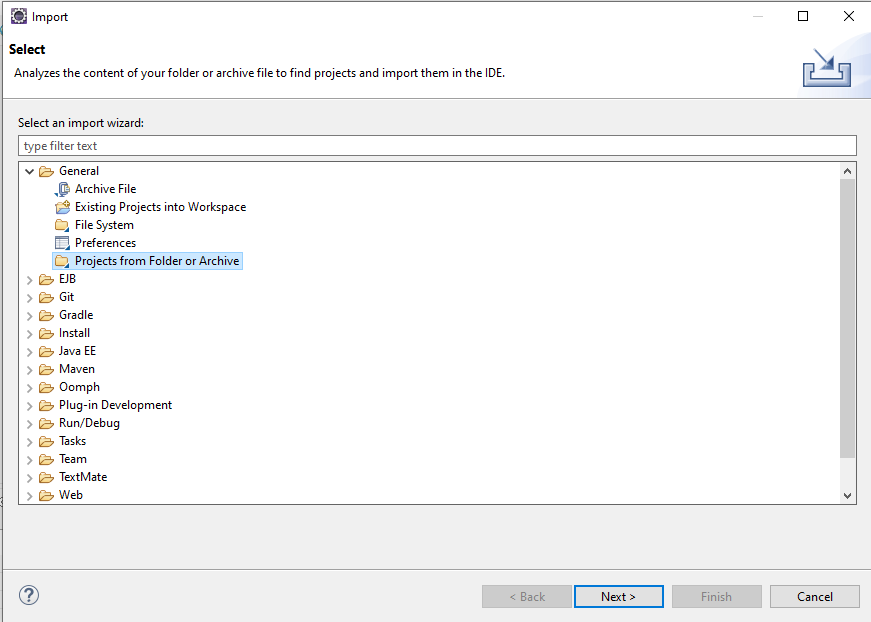
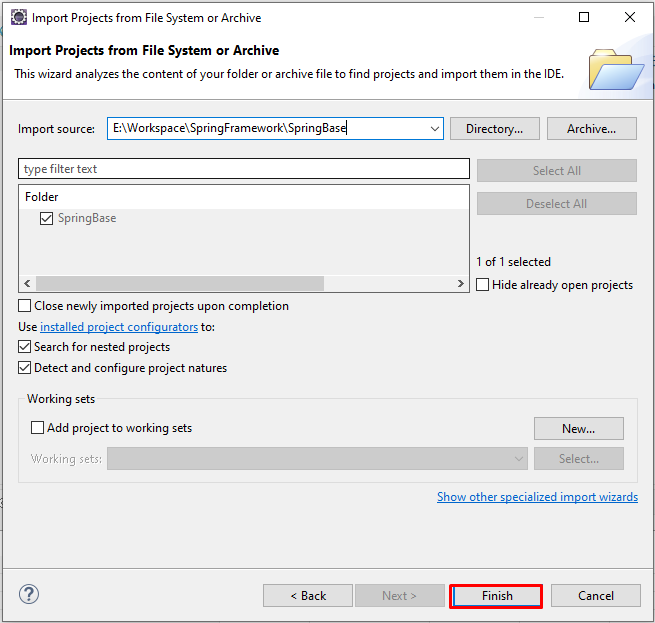
Thực hiện generate project với các thành phần module như hình



Hãy cài đặt Lombok cho IDE của mình.

## 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 |

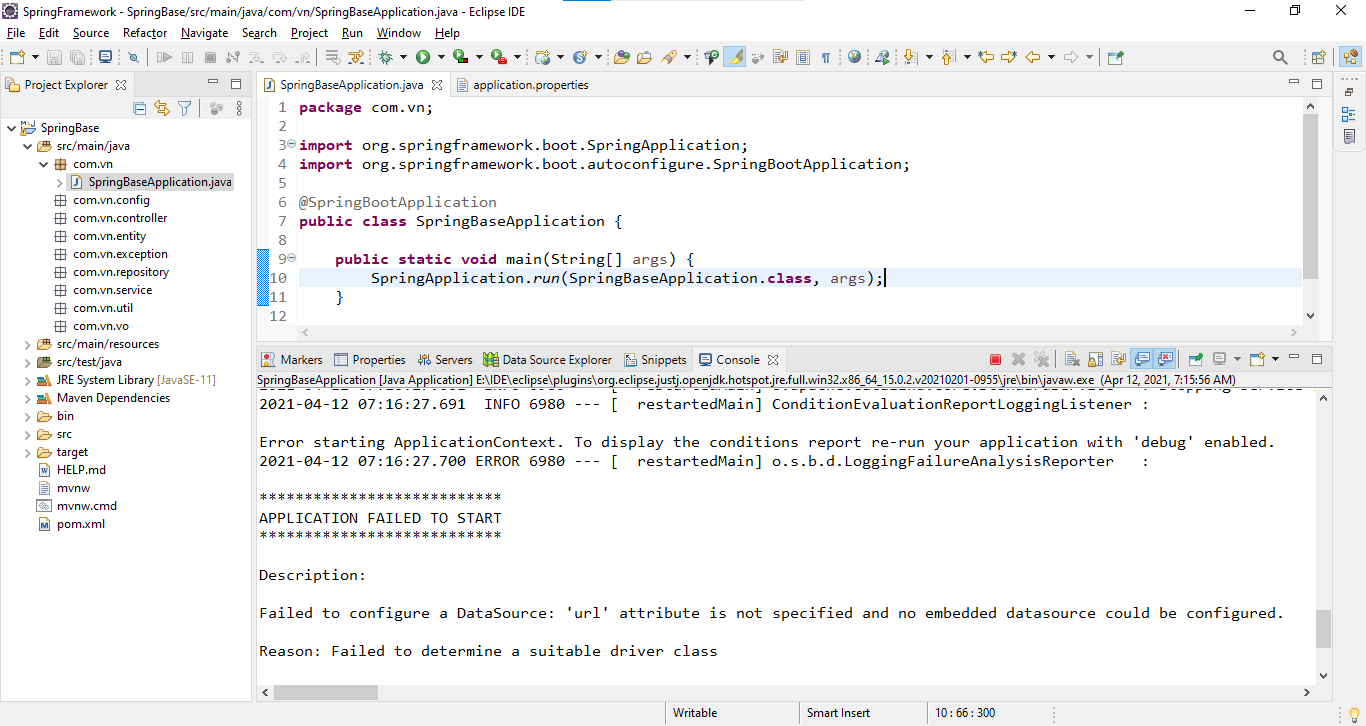
Bảng 1: Phân chia các packge cơ bản

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

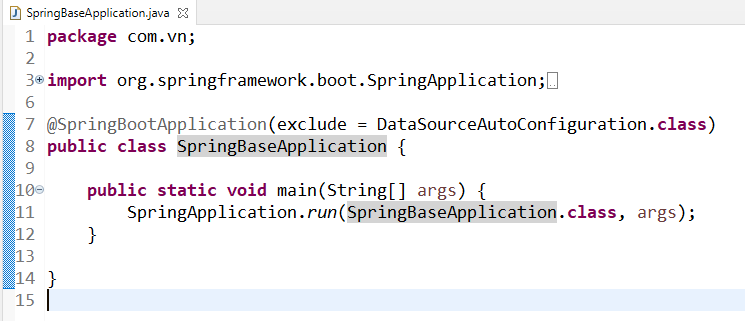
## Step3: Cấu hình Spring JPA application.properties

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

## Tạo Repository Layer

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