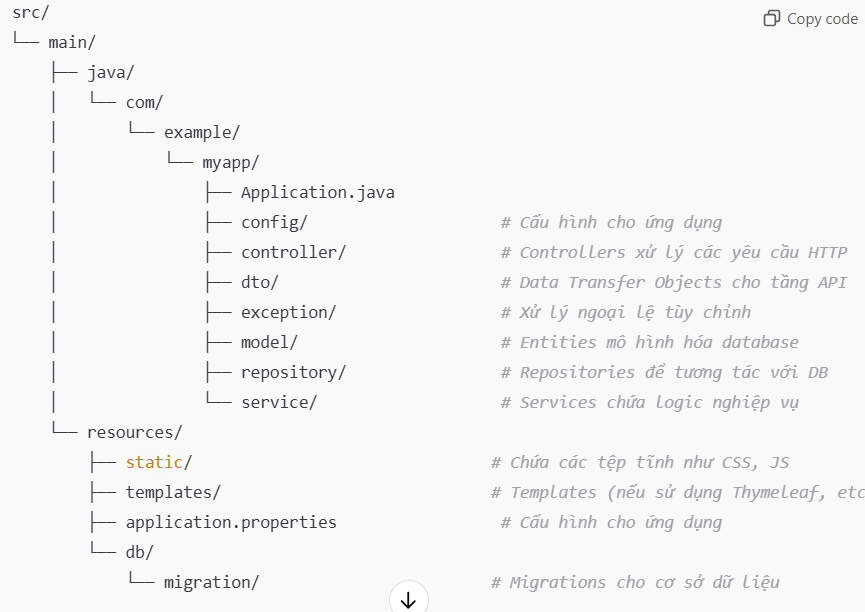
**JAVA SPRING BE**

1. Spring
   1. Hello world

package com.example.learn\_sping.controller;  
  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController //Khai báo đây là controller  
public class HelloWorldController {  
 @GetMapping("/") //Khai báo endpoint  
 public String helloWorld() {  
 return "Hello World";  
 }  
}

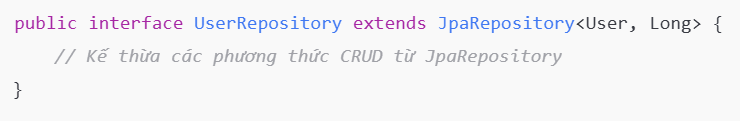
* 1. Tổ chức thư mục



* 1. Quy trình hoạt động
     1. Controller
* **Khởi đầu**: Quá trình xử lý bắt đầu từ controller, nơi các yêu cầu HTTP từ client được nhận. Controller trong Spring được đánh dấu bằng annotation @RestController hoặc @Controller.
* **Xử lý Yêu cầu**: Controller lấy dữ liệu từ yêu cầu, chẳng hạn như thông tin từ URI, tham số truy vấn, hoặc body của yêu cầu. Controller sau đó gọi một phương thức từ lớp service tương ứng để xử lý yêu cầu đó.
* **Lưu ý :** Có 1 khái niệm là DTO thực ra là model nhưng giúp tương tác ở controller, trong controller chúng ta sẽ không đụng tới cái entity hay domain model để giúp bảo mật
* @RestController
  @RequestMapping("/api/users")
  public class UserController {
      @Autowired
      private UserService userService;

      @GetMapping("/{id}")
      public ResponseEntity<UserDTO> getUserById(@PathVariable Long id) {
          UserDTO user = userService.getUserById(id);
          return ResponseEntity.ok(user);
      }
  }
  
  + 1. Service
* **Logic Nghiệp vụ**: Service là lớp nơi xử lý logic nghiệp vụ chính. Nó không quan tâm đến việc dữ liệu được gửi nhận như thế nào qua mạng, mà chỉ tập trung vào việc xử lý logic đó.
* **Giao tiếp với Repository**: Khi cần truy cập hoặc thay đổi dữ liệu trong cơ sở dữ liệu, service sẽ gọi đến repository.
* @Service
  public class UserService {
      @Autowired
      private UserRepository userRepository;

      public UserDTO getUserById(Long id) {
          User user = userRepository.findById(id).orElseThrow(() -> new NotFoundException("User not found"));
          return convertToDTO(user);
      }

      private UserDTO convertToDTO(User user) {
          // Logic để chuyển đổi từ User entity sang UserDTO
      }
  }
  
  + 1. Repository
* **Truy cập Dữ liệu**: Repository sử dụng Spring Data JPA để tương tác với cơ sở dữ liệu. Nó chứa các phương thức như findById(), findAll(), save(), delete(), v.v., mà các phương thức này tự động được Spring Data JPA hiện thực hóa.
* **Entity và Database**: Repository trực tiếp làm việc với các entity, là các đối tượng Java được ánh xạ tới các bảng trong cơ sở dữ liệu.
* 
  + 1. Model (Entity)
* **Ánh Xạ Dữ liệu**: Các entity là các class Java với các annotation JPA (@Entity, @Table, v.v.) để chỉ định cách chúng ánh xạ tới cơ sở dữ liệu. Các entity đại diện cho các bảng dữ liệu, và mỗi thực thể (instance) của entity đại diện cho một hàng trong bảng đó.
  1. Kết nối db và tạo api crud
     1. Thêm dependency

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
</dependency>  
<dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>8.0.33</version>  
</dependency>

* + 1. Thêm thông tin kết nối

spring.datasource.url=jdbc:mysql://localhost:3306/database\_name?useSSL=false&serverTimezone=UTC  
spring.datasource.username=root  
spring.datasource.password=  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.show-sql=true  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect

* + 1. Tạo model table

@Entity //Cho spring biết đây là 1 table
@Table(name = "User") //Name of table
@Getter
@Setter
public class User {
    @Id //Anotation define for id
    @GeneratedValue(strategy = GenerationType.IDENTITY) //Increase id, if we use UUID,it will create random id
    private Long id;

    @Column(nullable = false, unique = true)
    private String username;

    @Column(nullable = false)
    private String email;

    @Column(nullable = false)
    private String password;

    @ManyToMany(mappedBy = "participants")
    private Set<Conversation> conversations = new HashSet<>();

}
