



CSC12108 – ỨNG DỤNG PHÂN TÁN

HƯỚNG DẪN THỰC HÀNH

Xử lý nghiệp vụ trong microservice

I. Thông tin chung

Mã số:	HD01
Thời lượng dự kiến:	3 tiếng
Deadline nộp bài:	-
Hình thức:	-
Hình thức nộp bài:	-
GV phụ trách:	Phạm Minh Tú
Thông tin liên lạc với GV:	pmtu@fit.hcmus.edu.vn

II. Chuẩn đầu ra cần đạt

Bài hướng dẫn này nhằm mục tiêu đạt giúp sinh viên được các mục tiêu sau:

- Hiểu được mô hình xử lý logic trong kiến trúc microservice
- Sử dụng thư viện mongodb tích hợp Spring Boot

III. Mô tả

a. Môi trường phát triển

Trong bài này, các công cụ và môi trường dưới đây sẽ được dùng để hướng dẫn sinh viên thực hành môn học.

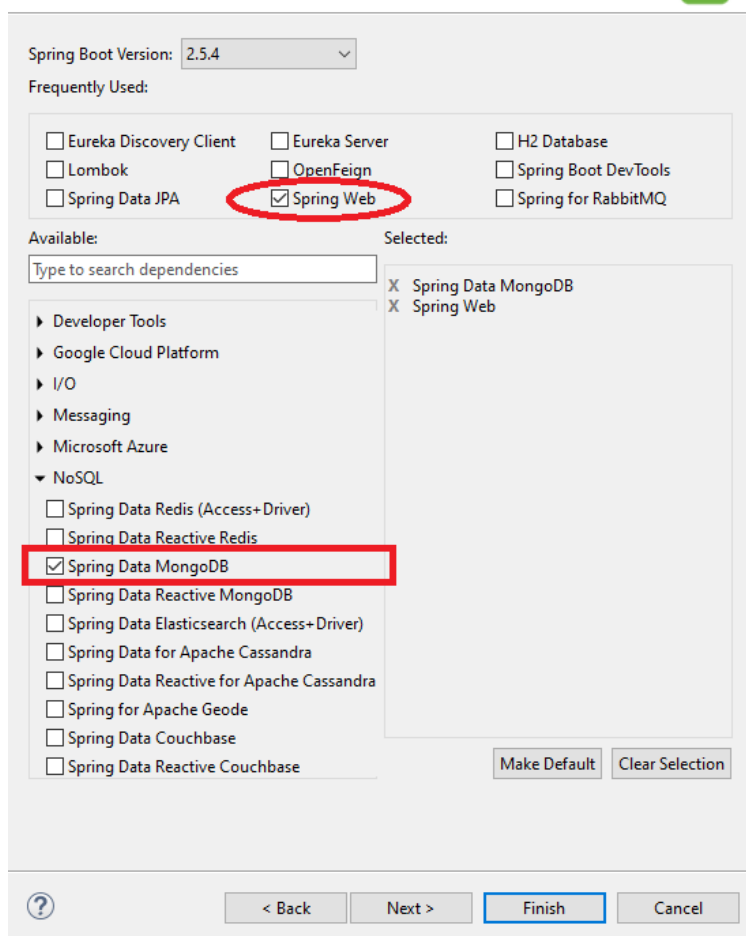
- **Eclipse IDE for Enterprise Java Developers.**
 - Version: 2019-03 (4.11.0)
 - Build id: 20190314-1200
- **Java JDK 1.8**

b. Cài đặt mongodb

Cài đặt theo hướng dẫn tại đây: <https://docs.mongodb.com/manual/installation/>

c. Tạo Project

Tạo project Spring Boot, chọn các thư viện như mongodb và web



Spring Boot Version: 2.5.4

Frequently Used:

- ☐ Eureka Discovery Client
- ☐ Eureka Server
- ☐ H2 Database
- ☐ Lombok
- ☐ OpenFeign
- ☐ Spring Boot DevTools
- ☐ Spring Data JPA
- ☒ Spring Web
- ☐ Spring for RabbitMQ

Available:

Type to search dependencies

- Developer Tools
- Google Cloud Platform
- I/O
- Messaging
- Microsoft Azure
- NoSQL
 - ☐ Spring Data Redis (Access+Driver)
 - ☐ Spring Data Reactive Redis
 - ☒ Spring Data MongoDB
 - ☐ Spring Data Reactive MongoDB
 - ☐ Spring Data Elasticsearch (Access+Driver)
 - ☐ Spring Data for Apache Cassandra
 - ☐ Spring Data Reactive for Apache Cassandra
 - ☐ Spring for Apache Geode
 - ☐ Spring Data Couchbase
 - ☐ Spring Data Reactive Couchbase

Selected:

- X Spring Data MongoDB
- X Spring Web

Make Default Clear Selection

< Back Next > Finish Cancel

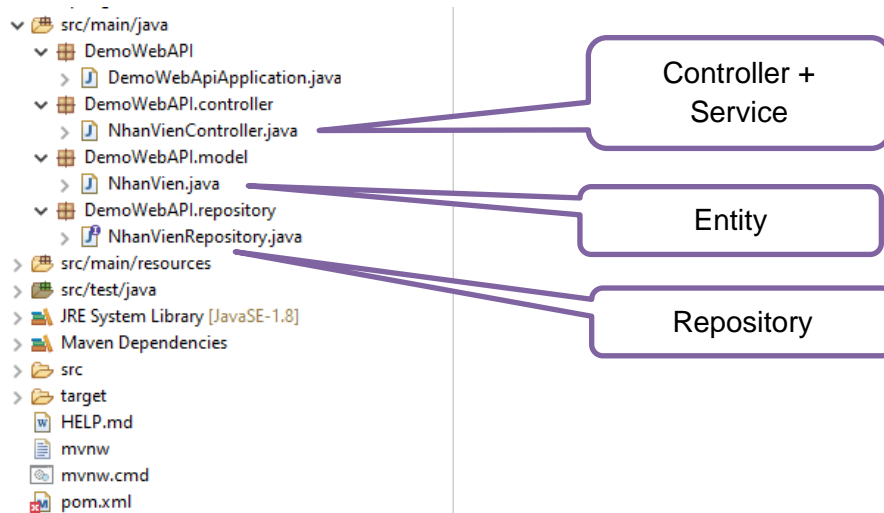
Thông thường một business logic bao gồm các thành phần sau:

Service: Cài đặt business của một lớp

Entity: Thực thể được ánh xạ cơ sở dữ liệu

Repository: Chịu trách nhiệm quản lý dữ liệu, tương tác trực tiếp với CSDL

Controller: Định nghĩa các API REST, được sử dụng bởi client.





Bởi vì ví dụ dưới đây không có xử lý business phức tạp, vì vậy để đơn giản chúng ta gộp controller và service lại.

Cấu hình thông tin kết nối mongodb

Resources/application.properties

```
spring.data.mongodb.database=quanlynhanvien  
spring.data.mongodb.port=27017
```

Entity

Ta định nghĩa **Entity** là lớp Nhân viên, bao gồm các thuộc tính đơn giản, bởi vì chúng ta sử dụng CSDL MongoDB nên dùng một số từ khóa ánh xạ giữa class và collection như dưới đây:

```
@Document(collection = "nhanvien")  
public class NhanVien {  
  
    @Id  
    private String manv;  
    private String hoten;  
    private int soNamKinhNghiem;  
    private boolean thuviec;  
  
    public String getManv() {  
        return manv;  
    }  
    public void setManv(String manv) {  
        this.manv = manv;  
    }  
    public String getHoten() {  
        return hoten;  
    }  
    public void setHoten(String hoten) {  
        this.hoten = hoten;  
    }  
    public int getSoNamKinhNghiem() {  
        return soNamKinhNghiem;  
    }  
    public void setSoNamKinhNghiem(int soNamKinhNghiem) {  
        this.soNamKinhNghiem = soNamKinhNghiem;  
    }  
    public boolean isThuviec() {  
        return thuviec;  
    }  
    public void setThuviec(boolean thuviec) {  
        this.thuviec = thuviec;  
    }  
    public NhanVien(String manv, String hoten, int soNamKinhNghiem, boolean  
thuviec) {  
        super();  
        this.manv = manv;  
        this.hoten = hoten;  
        this.soNamKinhNghiem = soNamKinhNghiem;  
        this.thuviec = thuviec;  
    }  
}
```



```
public NhanVien() {  
    super();  
}
```

```
}
```

Repository

Việc cài đặt Repository rất dễ dàng với thư viện spring boot mongodb, đặc biệt tính năng CRUD được thực hiện sẵn, chúng ta chỉ cần khai báo interface như bên dưới:

```
public interface NhanVienRepository extends MongoRepository<NhanVien, String>{  
  
    // CRUD  
  
}
```

Controller

Sau khi xây dựng Repository, cơ bản đã có thể quản lý dữ liệu (CRUD), tiếp theo định nghĩa các API để client truy cập và sử dụng.

Trong bài này, các API cơ bản để thêm, xóa, cập nhật và tìm kiếm thông tin nhân viên

Định nghĩa lớp controller cần **định nghĩa url** như sau:

```
@RestController  
@RequestMapping("/api")  
public class NhanVienController {  
  
}
```

http://localhost:8080/api

API **thêm** một nhân viên

```
@Autowired  
NhanVienRepository repo;
```

Sử dụng Repository để quản lý dữ liệu, bên dưới là API để thêm một nhân viên mới

```
@RestController  
@RequestMapping("/api")  
public class NhanVienController {  
  
    @Autowired  
    NhanVienRepository repo;  
  
    @PostMapping("/nhanvien")  
    public ResponseEntity<NhanVien> ThemNhanVien(@RequestBody NhanVien nhanvien) {  
        try {  
            NhanVien _nhanvien = repo.save(new NhanVien(nhanvien.getManv(), nhanvien.getHoten(),  
                nhanvien.getSoNamKinhNghiem(), nhanvien.isThuviec()));  
            return new ResponseEntity<>(_nhanvien, HttpStatus.CREATED);  
        } catch (Exception e) {  
            return new ResponseEntity<>(null, HttpStatus.INTERNAL_SERVER_ERROR);  
        }  
    }  
  
    ...  
}
```

POST: http://localhost:8080/api/nhanvien

API **cập nhật** một nhân viên dựa vào Id của nhân viên



```
@RestController
@RequestMapping("/api")
public class NhanVienController {

    @Autowired
    NhanVienRepository repo;

    @PutMapping("/nhanvien/{id}")
    public ResponseEntity<NhanVien> CapNhatNhanVien(@PathVariable("id") String id, @RequestBody NhanVien
nhanvien) {

        Optional<NhanVien> nhanvienData = repo.findById(id);

        if (nhanvienData.isPresent()) {
            NhanVien _nhanvien = nhanvienData.get();
            _nhanvien.setHoten(nhanvien.getHoten());
            _nhanvien.setSoNamKinhNghiem(nhanvien.getSoNamKinhNghiem());
            _nhanvien.setThuviec(nhanvien.isThuviec());
            return new ResponseEntity<>(repo.save(_nhanvien), HttpStatus.OK);
        } else {
            return new ResponseEntity<>(HttpStatus.NOT_FOUND);
        }
    }
}

....
}
```

PUT: <http://localhost:8080/api/nhanvien/12345>

Tương tự cho các API xóa và tìm kiếm nhân viên

```
@DeleteMapping("/nhanvien/{id}")
public ResponseEntity<HttpStatus> XoaMotNhanVien(@PathVariable("id") String id) {
    try {
        repo.deleteById(id);
        return new ResponseEntity<>(HttpStatus.NO_CONTENT);
    } catch (Exception e) {
        return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}

@DeleteMapping("/nhanvienall")
public ResponseEntity<HttpStatus> XoaTatCaNhanVien() {
    try {
        repo.deleteAll();
        return new ResponseEntity<>(HttpStatus.NO_CONTENT);
    } catch (Exception e) {
        return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
    }
}

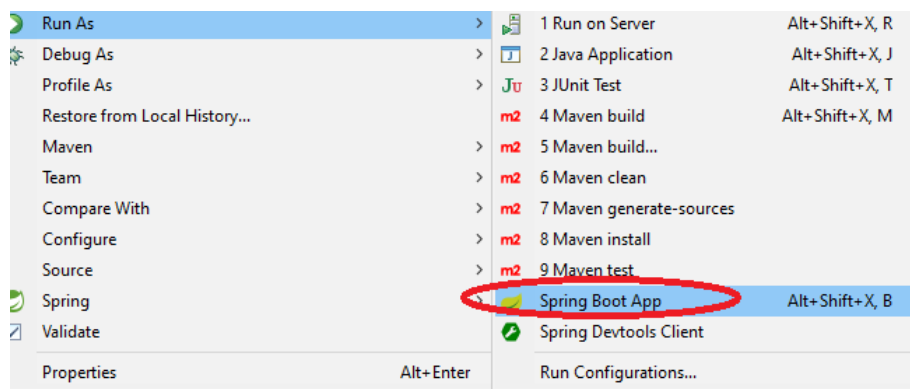
@GetMapping("/nhanvien")
public ResponseEntity<List<NhanVien>> XemDanhSachNhanVien() {
    try {
        List<NhanVien> nhanvienlst = new ArrayList<NhanVien>();

        repo.findAll().forEach(nhanvienlst::add);

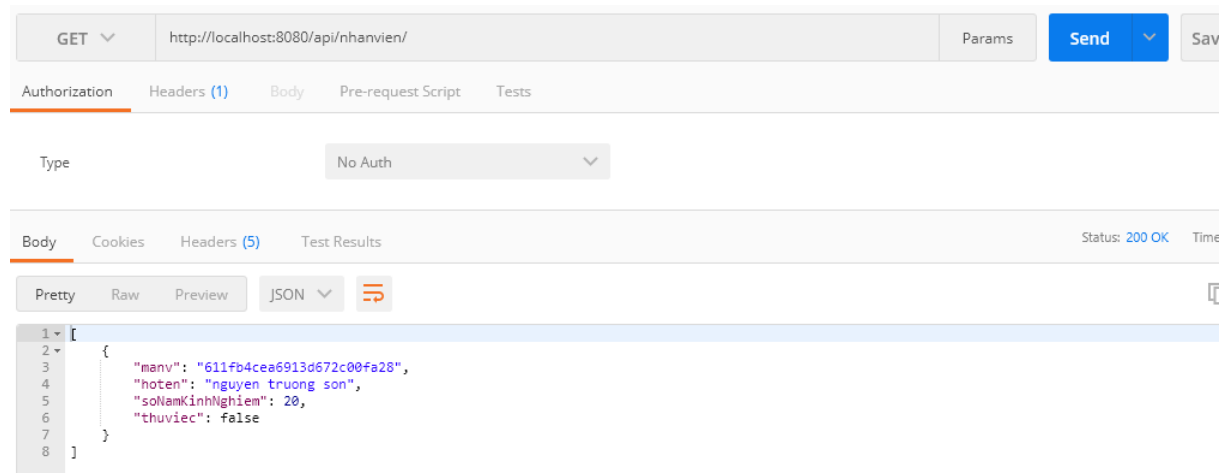
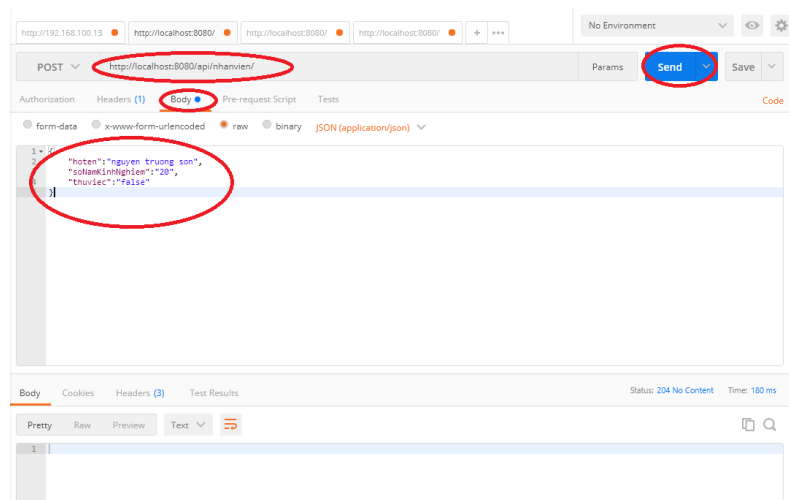
        if (nhanvienlst.isEmpty()) {
            return new ResponseEntity<>(HttpStatus.NO_CONTENT);
        }

        return new ResponseEntity<>(nhanvienlst, HttpStatus.OK);
    } catch (Exception e) {
        return new ResponseEntity<>(null, HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

Cuối cùng triển khai project tại localhost



Kiểm tra các API thông qua công cụ **postman**





The screenshot displays two REST client interfaces. The top interface shows a PUT request to `http://localhost:8080/api/nhanvien/611fb4cea6913d672c00fa28` with a JSON body: `{ "hoten": "nguyen truong son", "soNamKinhNghiem": "15", "thuviec": "false" }`. The bottom interface shows a DELETE request to `http://localhost:8080/api/nhanvienall` with a status of 204 No Content.

PUT Request:

- Method: PUT
- URL: `http://localhost:8080/api/nhanvien/611fb4cea6913d672c00fa28`
- Body:

```
{
  "hoten": "nguyen truong son",
  "soNamKinhNghiem": "15",
  "thuviec": "false"
}
```

DELETE Request:

- Method: DELETE
- URL: `http://localhost:8080/api/nhanvienall`
- Status: 204 No Content

IV. Tài liệu tham khảo

<https://docs.mongodb.com/manual/installation/>

V. Các quy định khác