TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH KHOA CÔNG NGHỆ THÔNG TIN

--------------------------------

Trần Quang Minh- 1451010108

**KẾT NỐI APPLICATION VỚI**

**BIG DATA THÔNG QUA**

**SPRING DATA JPA**

# BÁO CÁO THỰC TẬP TỐT NGHIỆP

Ngành: Công Nghệ Thông Tin

Hướng chuyên ngành: Cơ sở dữ liệu

Lớp: DH14TH02

NGƯỜI HƯỚNG DẪN  
Ths.Võ Thị Hồng Tuyết

Thành phố Hồ Chí Minh – Năm 2017

**LỜI CẢM ƠN**

Để hoàn thành báo cáo thực tập em đã nhận được nhiều sự giúp đỡ quý báu của quý thầy cô tại Trường Đại học Mở TP Hồ Chí Minh Khoa Công Nghệ Thông Tin.

Em xin chân thành cám ơn các thầy cô trong Khoa Công Nghệ Thông Tin đã tạo điều kiện cho chúng em thực hiện báo cáo thực tập này. Các thầy cô đã tận tình cung cấp, giảng dạy cho chúng em những kiến thức cơ bản, để từ đó có thể triển khai vào đề tài.

Đặc biệt em xin gửi lời cảm ơn sâu sắc tới cô Võ Thị Hồng Tuyết đã nhiệt tình hướng dẫn em hoàn thành báo cáo thực tập này.

Em xin chân thành cám ơn.

TP. Hồ Chí Minh, 26/11/ 2017

Trần Quang Minh- DH14TH02

**ĐÁNH GIÁ KẾT QUẢ THỰC TẬP CỦA KHOA**

**NHẬN XÉT CỦA CƠ QUAN THỰC TẬP**

**MỤC LỤC**

[**Chương 1. NỘI DUNG BÁO CÁO THỰC TẬP** 1](#_Toc500391184)

[**1.1 Lịch làm việc nơi thực tập** 1](#_Toc500391185)

[**1.2 Giới thiệu về tổ chức của nơi thực tập** 1](#_Toc500391186)

[**1.3 Nội dung nhiệm vụ chính được giao** 1](#_Toc500391187)

[**Chương 2. NỘI DUNG CÔNG VIỆC VÀ KẾT QUẢ ĐẠT ĐƯỢC** 2](#_Toc500391188)

[**2.1 Đặt vấn đề** 2](#_Toc500391189)

[**2.1.1 Nhu cầu thị trường** 2](#_Toc500391190)

[**2.1.2 Những khó khăn** 2](#_Toc500391191)

[**2.2 Giới thiệu đề tài** 2](#_Toc500391192)

[**2.3 Nội dung đề tài** 4](#_Toc500391193)

[**2.3.1 Spring Core** 4](#_Toc500391194)

[**2.3.2 Spring Expression Language (SpEL)** 5](#_Toc500391195)

[**2.3.3 Spring Boot** 5](#_Toc500391196)

[**2.3.4 Java Persistence API (JPA)** 7](#_Toc500391197)

[**2.3.5 JPA- Thành phần Object Relational Mapping (ORM Component)** 9](#_Toc500391198)

[**2.3.6 Java Persistence Query Language (JPQL)** 11](#_Toc500391199)

[**2.3.7 Spring Data JPA** 12](#_Toc500391200)

[**2.4 Tổng kết đề tài** 17](#_Toc500391201)

[**Chương 3. KẾT QUẢ ĐẠT ĐƯỢC QUA ĐỢT THỰC TẬP** 18](#_Toc500391202)

[**3.1 Những kiến thức lý thuyết đã được cũng cố** 18](#_Toc500391203)

[**3.2 Những kỹ năng thực thành đã được học thêm** 18](#_Toc500391204)

[**3.2.1 Maven project** 18](#_Toc500391205)

[**3.2.2 Design Parttern** 18](#_Toc500391206)

[**3.3 Những kinh nghiệm đã được tích lũy** 19](#_Toc500391207)

**DANH MỤC HÌNH ẢNH**

*Hình 2.1 Mô hình tổng quan về đề tài 3*

*Hình 2.2 Mô hình hoạt động của IoC Container 5*

*Hình 2.3 Mô hình tổng quan về Spring Boot 6*

*Hình 2.4 Mô hình cấu tạo Spring Boot 7*

*Hình 2.5 Mô hình tổng quan về JPA 8*

*Hình 2.6 Mô hình hoạt động của JPA thông qua O\R Mapping 10*

*Hình 2.7 Mô hình cách kết nối với Database theo JPA 12*

*Hình 2.8 Mô hình tổng quan về Repository interface của Spring Data JPA 13*

*Hình 2.9 Mô hình hoạt động của Spring Data JPA 15*

1. **NỘI DUNG BÁO CÁO THỰC TẬP**
   1. **Lịch làm việc nơi thực tập**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thứ 2 | Thứ 3 | Thứ 4 | Thứ 5 | Thứ 6 |
| Cả ngày (từ 8h – 18h) | Cả ngày (từ 8h – 18h) | Cả ngày (từ 8h – 18h) | Buổi chiều (từ 13h- 18h) | Cả ngày (từ 8h – 18h) |

* 1. **Giới thiệu về tổ chức của nơi thực tập**

Công ty Swiss Post Solutions Ltd. là công ty TNHH 100% vốn đầu từ trực tiếp nước ngoài, với những dịch vụ số hóa dữ liệu, xử lý dữ liệu, xử lý hình ảnh, phát triển phần mềm, các dịch vụ công nghệ thông tin, dịch vụ điều hành IT, làm việc 24/7, 365 ngày.

Bảo mật thông tin: Nghiêm túc tuân thủ đầy đủ, đúng Chỉ thị 95/46/EC của Liên minh Châu Âu và Luật Bảo mật Dữ liệu của Cộng hoà Liên bang Đức (BDSG) [6]

[*Chính sách bảo mật thông tin ISMS*](http://spsvietnam.vn/wp-content/uploads/2017/10/ISMS_Information_Security_Policy_V3.3_A.5_scan.compressed.pdf)

Chất lượng vượt trội: Cam kết cung cấp các dịch vụ với chất lượng tốt nhất theo tiêu chuẩn của những đối tác mà chúng tôi hợp tác.

Xây dựng những mối quan hệ bền vững với nhân viên: Phấn đấu tạo ra môi trường làm việc thân thiện, nơi mà nhân viên cảm thấy mình đang được sống chung một gia đình và được tự do phát biểu những ý tưởng của họ.

Với mục đích tăng trưởng liên tục, chúng tôi luôn có những ưu tiên như sau: doanh thu, cải tiến, thị trường mới, nhân viên và cơ sở kinh doanh, lợi nhuận.

* 1. **Nội dung nhiệm vụ chính được giao**
* Tìm hiểu về : Spring core framework , Sping data JPA, Sping Boot.
* Tìm hiểu giải pháp lưu postgre version 10 theo kiểu JSON (JSON type) như: thêm, xóa, sửa, tìm kiếm, và hiệu năng của nó so với lưu kiểu bảng (Table type).

1. **NỘI DUNG CÔNG VIỆC VÀ KẾT QUẢ ĐẠT ĐƯỢC**
   1. **Đặt vấn đề**
      1. **Nhu cầu thị trường**

Hiện nay công nghệ ngày càng phát triển, nhu cầu của nhà đầu tư, người sử dụng ngày càng cao. Và bất cứ ứng dụng nào dù nhỏ hay lớn đều cần lưu trữ thông tin, vì nhu cầu đó mà các nhà phát triển phát hành ra rất nhiều các loại cơ sở dữ liệu (database) với ưu và nhược điểm riêng.

* + 1. **Những khó khăn**

Mỗi cơ sở dữ liệu (database) đều có cách kết nối và sử dụng riêng, nếu sử dụng cách cấu hình theo kiểu cũ (JDBC, …) sẽ dẫn đến việc gây khó khăn cho các nhà phát triển ứng dụng. Những các kết nối cũ sẽ phải ánh xạ rất nhiều trường trong một table, rất khó khăn khi muốn thay đổi các trường trong bảng (table).Và phải phụ thuộc vào các câu query trong SQL hoặc NoSQL, không linh động trong việc chuyển đổi giữa SQL và NoSQL.

Vì lý do đó ta cần một công nghệ có thể linh động trong việc kết nối với cơ sở dữ liệu (database). Và đó chính là hệ sinh thái Spring framework.

* 1. **Giới thiệu đề tà**

Hệ sinh thái Spring framework là một framework rất lớn gồm các framework nhỏ và trong đó có Spring Data JPA Framework sẽ giải quyết tất cả những vấn đề nêu trên.[1]

Another Layer

Postgres

Spring Boot

Mongo BD

MySQL

Spring Data JPA

*Hình 2.1 Mô hình tổng quan về đề tài*

Với Spring Data JPA chúng ta không còn phải nhọc công trong việc ánh xạ (mapping) từng cột với từng thuộc tính (attribute) của một đối tượng (Obejct) mà thay vào đó mọi công việc như tạo bảng (table), tạo câu query , ánh xạ (mapping) từng cột đều được tự động hóa.

Như vậy với một cơ sở dữ liệu (database) lớn với một đối tượng (object) có rất nhiều thuộc tính, chúng ta sẽ tiết kiệm được rất nhiều thời gian cũng như công sức để kết nối giữa application với cơ sở dữ liệu (database).

* 1. **Nội dung đề tài**

Trước khi nói đến các giải pháp để giải quyết các vấn đề về cơ sở dữ liệu (database), chúng ta cần làm rõ các tiêu chuẩn cho Spring framework và JPA đề ra.

* + 1. **Spring Core**

Trước đây khi muốn tạo ra một đối tượng (Object) ta cần sử dụng new để gọi constructor khởi tạo một đối tượng Object. Trong Spring sẽ loại bỏ hoàn toàn khái niệm new một Object, thay vào đó Spring đưa ra 2 khái niệm chính là Inversion of Control (IoC) và Dependency Injection (DI). Đây được coi là phần cốt lõi mà bất cứ Spring framework nào cũng sử dụng. [2]

* Inversion of Control (IoC)

Inversion of Control có nghĩa là đảo ngược sự điều khiển. Theo cách truyền thống một đối tượng được tạo ra từ 1 class, các trường (field) của nó sẽ được gán chính bên trong class đó. Spring đã làm ngược lại với cách truyền thống đó, các đối tượng (object) được tạo ra và một vài trường (field) của nó được inject giá trị bên ngoài bởi một cái được gọi là IoC.

IoC Container là bộ chứa tất cả các Spring BEAN được sử dụng trong ứng dụng.IoC Container đại diện cho org.springframework.context.ApplicaitonContext, nó chịu trách nhiệm cho việc khởi tạo, cấu hình và quá lý các beans.

* Dependency Injection (DI)

Dependency Injecction là một quá trình mà theo đó đối tượng (object) định nghĩa sự phụ thuộc của nó, sự định nghĩa đó là những đối tượng (object) khác mà chúng làm việc với. Chúng làm việc với nhau chỉ thông qua constructor arguments , hoặc set method trên (đối tượng) object instance (thể hiện của đối tượng) sau khi nó được khởi tạo. Container sẽ tự động inject sự phụ thuộc khi nó khởi tạo các Bean. Đây là một quá trình về cơ bản là một sự đảo ngược, vì thế nên nó có tên là Invertion of Control (IoC).

Dependency Injection là một khái niệm hợp lệ với bất kỳ một ngôn ngữ chương trình nào. Nó là một khái niệm tổng quát được gọi bởi Inversion of Control (IoC).

* Bean

Bean là các Java Object mà từ đó tạo nên khung sườn của một Spring application. Chúng được khởi tạo, cấu hình và quản lí bởi IoC Container. Bean được khởi tạo bằng cách thông qua cấu hình XML file , Anntation , Java-based. [2]

Service

Repository

inject

inject

Bean

My Service

My Repository

IoC Container

*Hình 2.2 Mô hình hoạt động của IoC Container*

* + 1. **Spring Expression Language (SpEL)**

Là một ngôn ngữ biểu thức mạnh mẽ do Spring framework định nghĩa, hỗ trợ query và thao túng một sơ đồ đối tượng (object) tại runtime, có thể sử dụng cho toàn hệ sinh thái Spring framework.[2]

SpEL được tạo nên vói mục đích giúp tạo ra một nguyên tắt giao tiếp chung cho toàn bộ các úng dụng sử dụng Spring framework, qua đó có thể dễ dàng mở rộng thêm các ứng dụng khi cần.

* + 1. **Spring Boot**

Spring Framework là một project rất lớn bao gồm các framework nhỏ hơn bên trong. Như Spring Data Framework, Spring Web MVC Framework, Spring Batch Framework, Spring Cloud Framework, ….Mỗi Framework đều có một version riêng, vì vậy, khi kết hợp và lồng nhiều framework với nhau, rất khó để tìm một phiên bản (version) chung cho tất cả các Framework. Và dùng nhiều các Framework cũng sẽ dẫn đến một khối lượng beans phải cấu hình rất lớn, số XML file cũng vì thế mà tăng lên, chồng chéo lẫn nhau, rất khó có lập trình viên quản lí các Beans cũng như tìm version để các framework tương thích với nhau. Vì vậy Spring Boot ra đời để giải quyết các yếu điểm này.

* Khái niệm Spring Boot

Spring boot là một project nằm trên tầng IO Execution (Tầng thực thi) của Spring IO Framework.

Spring boot là một bước tiếp theo của Spring, để làm cho Spring dể dàng hơn trong việc thiết lập và phát triển ứng dụng. Với Spring Boot việc cấu hình sẽ giảm thiểu tối đa. Spring boot hỗ trợ bộ nhúng (Embedded containers) điều này cho phép các ứng web có thể chạy độc lập mà không cần phải khai báo lên các Web Server. [3]

* Mô hình Spring boot đối với Database:

IO Exection

Spring Booot

Big Data

Relational Data Access

Non-Relational Data Access

*Hình 2.3 Mô hình tổng quan về Spring Boot*

* Ưu điểm của Spring Boot:

Embedded HTTP Servers

XML <bean> Configuration

Spring Framework

Spring Boot

*Hình 2.4 Mô hình cấu tạo Spring Boot*

Với Spring boot ta sẽ loại bỏ hoàn toàn các XML files và sẽ thay thế nó bằng các anntation(@Service, @Companet, @Controller, …) giúp tránh việc phải code nhiều,

các Beans sẽ được inject bằng cách sử dụng @Autowired annotation.

Spring Boot định nghĩa ra một application.properties file để giúp cấu hình mọi thứ một cách tự động.

* + 1. **Java Persistence API (JPA)**
* Giới thiệu JPA

Java Persistence API hay JPA là một đặc tả Java cho việc ánh xạ giữa các Java Object với cơ sở dự liệu (database) sử dụng công nghệ phổ biến là ORM(Object Relational Mapping). JPA API cung cấp đầy đủ các công cụ cho phép người lập trình có thể tạo cơ sở dữ liệu (database) một cách đơn giản và nhanh chóng.

JPA ra đời cùng với chuẩn J2EE 5, JPA có thể sử dụng cho cả ứng dụng J2EE và J2SE, hỗ trợ cấu hình bằng cả annotation và xml file. [5]

* Kiến trúc JPA

JPA API

Entity Manager Factory

Entiry Transaction

Entity

Query

Entity Maganer

Persistence

Database

*Hình 2.5 Mô hình tổng quan về JPA*

* EntityManagerFactory: Đây là một factory class của Entity Manager, nó khởi tạo và quản kí nhiều thể hiện của Entity Manager
* Entity Manager: là một interface cung cấp các API cho việc tương tác với các Entity, nó làm việc như một factory cho thể hiện của Query. Chức năng cơ bản của Entity Manager như:
  + - Persist: Phương thức này dùng để lưu một thực thể mới tạo vào cơ sở dữ liệu (database)
    - Merge: Dùng để cập nhật trạng thái của entity vào cơ sở dữ liệu (database)
    - Remove: Xóa một thể hiện của entity
* Enity là các Persistence Object thể hiện một mẫu tin trong database. Entity chỉ là các lớp POJO (Plain Old Java Object) đơn giản, dể dàng tạo. Một số đặc điểm của Entity:
  + - Entity có thể tương tác với cơ sở dự liệu (database)
    - Entity được xác định thông qua một định danh (tương đương với khóa chính trong table của database)
    - Entity hỗ trợ transaction
    - Entity hỗ trợ kế thừa giống với các lớp thường khác
    - Entity có thể được khai báo với annotation (@Entity) và một entity phải có khóa chính (@Id)
* Entity Transaction: Nó có quan hệ 1-1 với Entity Manager. Cho mỗi Entity Manager, sự họa động được duy trì bởi Entity Transaction class.
* Persistence: Đây là class chứa các hàm static to có được Entity Manager Factory
* Query: Đây là interface được implemented bởi mỗi JPA để có được mối quan hệ đối tượng (object) mà đáp ứng các tiêu chí.
* Lợi ích

Chúng giúp người lập trình bằng cách giảm việc phải code nhiều lưu trữ xuống dữ liệu xuống cơ sở dữ liệu (database). Vì vậy, lập trình viên chỉ cần phải tập trung vào việc viết code để ánh xạ các classes với bảng trong cơ sở dữ liệu (database table).

* + 1. **JPA- Thành phần Object Relational Mapping (ORM Component)**
* Giới thiệu ORM

Object Relational Mapping là một chương trình có khả năng chuyển đổi dữ liệu từ dạng đối tượng (object) sang dạng relational type và ngược lại. [5]

Các tính năng chính của ORM là ánh xạ hoặc ràng buộc một đối tượng vào dữ liệu của nó trong cơ sở dữ liệu (database). Trong khi ánh xạ (mapping) cần phải xem xét data, loại data và các mối quan hệ của nó so với thực tế.

* Mô hình ORM

Relational

Object Gird

Object

JPA Loader

Entity

Mapping

Database

Mapping file

JPA provider

DAO/Service

*Hình 2.6 Mô hình hoạt động của JPA thông qua O\R Mapping*

* Bước một:

Các POJO classes, service interface và các classes đây là thành phần business chính. Ví dụ: Employee POJO class chứa các thuộc tính như là id, name, salary, designation và các hàm như get và set của các thuộc tính. Employee DAO/Service classes chứa các hàm service như là create employee, find employee, and delete employee.

* Bước hai:

JPA Provider: Là các framework mà hỗ trợ JPA (javax.persistence) như là Ecliplink, Hibernate, …

Mapping file: ORM.xml chứa cấu hình ánh xạ giữa data trong một POJO class và data trong một quan hệ (relational) database

JPA Loader: làm việc như một bộ nhớ cache, nó hoạt động như một bản copy của cơ sở dữ liệu (database) để tương tác với service class (create, update, insert, delete) của POJO data (thuộc tính của POJO class).

Object Grid: là một vị trí tạm thời có thể lưu trữ bản copy của mối quan hệ data. Tất cả các queries đều được thực thi trên bản data copy này và nếu không có phát sinh lỗi nó sẽ commit xuống cơ sở dự liệu (database). Chỉ sau khi được commited xuống cơ sở dữ liệu (database) thì database mới thực sự bị ảnh hưởng. Giai đoạn này còn gọi là Transaction.

* Bước ba:

Nó chứa các mối quan hệ data (relational data) được kết nối logic đến các thành phần business. Chỉ khi các thành phần business commit data, nó mới được lưu trữ vào data vật lý.

Toàn bộ cơ chế tác động chương trình trên được gọi là Object relational mapping.

* + 1. **Java Persistence Query Language (JPQL)**

Java Persistence Query Language là ngôn ngữ được định nghĩa trong JPA. Nó được sử dụng để tạo ra query đối với entities để lưu trữ trong một mối quan hệ cơ sở dữ liệu (database). JPQL được phát triển trên các ký hiệu SQL cơ bản. Nhưng nó không ảnh hưởng trực tiếp đến cơ sở dữ liệu (database).

JPQL có thể lấy thông tin hoặc data bằng cách sử dụng “SELECT”, có thể cập nhật một số lượng lớn thông tin bằng cách sử dụng “UPDATE” và “DELETE”. EntityManager.createQuery() API sẽ hỗ trợ cho ngôn ngữ query này.

* Mô hình JPA kết nối với Database (O/R Mapping)

Database

JPA

O\R Mapping và Persistence

Entity

Column2

Column1

id

Application Modules

Private long id;

Table

Private type name

Priavte type name

*Hình 2.7 Mô hình cách kết nối với Database theo JPA*

* Cấu trúc Query

JPQL rất giống các ký hiệu của SQL, nhưng SQL làm việc trực tiếp với cơ sở dữ liệu (database) (column và table), trong khi JPQL làm việc với Java class và instance của Java class.

Ví dụ: để select một cột trong bảng ta dùng :

* SQL - SELECT name Column FROM MyTable
* JPQL- SELECT myTable FROM MyTable myTable
  + 1. **Spring Data JPA**

Với những phần trình bày trên tôi đã giới thiệu những khái niệm của những cơ chế để giúp làm việc với Spring Data JPA. Vậy câu hỏi được đặt ra là những cơ chế trên liên quan như thế nào đến Spring Data JPA và vì sao Spring Data JPA là giải pháp cho vấn đề trên?

* + Giới thiệu Spring Data JPA

Spring Data JPA không phải là một JPA provider, nó chỉ là một thư viện hoặc một framework được thêm vào như là một phần mở rộng của JPA provider. Phần quan trọng nhất của Spring data JPA là các interface được kế thừa từ các repository interface cho Spring Data JPA định nghĩa. [4]

* + Spring Data Repository

Toàn bộ sức mạnh của Spring data JPA nằm trong các repository interface được cung cấp bởi Spring Data Commons project. Và ta vẫn có thể làm việc với repository interface một cách bình thường bằng cách tạo ra các interface và kế thừa repository interface

Spring Data JPA

Spring Data Commons

QueryDslPredicateExecutor<T>

JpaSpecificationExecutor<T>

JpaRepository<T, ID extends Serializable>

PagingAndSortingRepository<T, ID extends Serializable>

CrudRepository<T, ID extends Serializable>

Repository<T, ID extends Serializable>

*Hình 2.8 Mô hình tổng quan về Repository interface của Spring Data JPA*

Repository<T, ID extends Serializable> : là một interface có hai mục đích :

* + Nắm bắt, quản lí các entities và kiểu dữ liệu id của entity đó.
  + Nó giúp Spring Container để tìm ra các repository interface cụ thể trong suốt quá trình quét (scan) classpath

CrudRepository<T, ID extends Serializable>: là interface cung cấp các phương thức CRUD (Create , Read, Update, Delete) cho việc quản lý entity

CrudRepository<T, ID extends Serializable>: là interface khai báo các phương thức hổ trợ việc sắp xếp, phân trang dữ liệu từ cơ sở dữ liệu (database).

JpaRepository<T, ID extends Serializable>: là interface JPA repository cụ thể.

* Mô hình của Spring Data JPA

Service

Repository

Repository Implemention

Application Modules

JPA

O/R Mapping

Spring Data JPA

JPA Provider

Data Source

JDBC

Interfaces

JDBC Basic APIs

Configuration for Connects

JDBC Implemention

JDBC Driver

Persistence Layer

Database

Oracle

Postgres

MySQL

*Hình 2.9 Mô hình hoạt động của Spring Data JPA*

* Cơ chế hoạt động của Spring Data JPA

Một cơ sở dữ liệu (database) nào cũng cần những chức năng cơ bản như: tìm kiếm (SELECT), cập nhật (UPDATE), xóa (DELETE), khởi tạo (CREATE). Và với JPA ta đã được cung cấp cách ánh xạ một đối tượng (object) thành một table dưới database và cách giao tiếp với database, nhưng yếu điểm của JPA là vẫn quá tốn nhiều thời gian vào việc tạo các Crud class cho các Entitties. Spring Data JPA ra đời để cải tiến đều đó.

Spring Data JPA hoạt động bằng cách các lập trình viên sẽ tạo ra một repository interface có kiểu của các Entities và kế thừa các reopository interface đã được Spring Data JPA định nghĩa sẵn.

Ví dụ: public interface MyRepository extends CrudRepository<MyEntity, Long>{}

Spring Data JPA sẽ tìm kiếm tất cả interface đã kế thừa từ các repository interface đã định nghĩa sẵn và tự động implement nó.

Lúc này Spring data sẽ gọi đến JPA APIs (Application Programming Interface) và tự động implements nó theo JPA Provider (có thể là Hibernate, Ecliplink,…) để kết nối với Database.

Ở đây nếu chúng ta không sử dụng Spring Boot thì bắt buộc phải cấu hình Data Source cho Spring Data JPA để nó có thể kết nối với cơ sở dữ liệu (database). Spring Boot có thể tự động cấu hình tất cả mọi thứ và chỉ cần cấu hình thông qua application.properties file.

JPA sẽ ánh xạ giữa các thuộc tính trong entity thành các cột trong bảng (table) của cơ sở dữ liệu (database).

Đồng thời nó sẽ tự động pasre tên của các phương thức trong MyRepository interface thành các câu query (Update, Create , Delete, Select, …) để truy xuất dữ liệu theo chuẩn Spring Expression Language (SpEL). Ví dụ: public interface MyRepository extends CrudRepository<MyEntity, Long>{

MyEntity findMyEntityById(long id);

}

Và tất cả những cơ chế trên sẽ được tự động cấuu hình và điều khiển bởi Spring Boot.

* Query của Spring Data JPA

Để có thể pasre tên của các phương thức trong repository interface, Spring Data JPA đã dự vào Spring Expression Language và định nghĩa một nguyên tắc cho cách đặt tên của các phương thức:

Tên của các phương thức trong repository interface phải là một trong những từ sau: find…By…, read…By…, query…By…, count…By… và get…By

Phải thiết lập kiểu trả về cho phương thức.

Ngoài ra còn có thể giới hạn số lượng bằng cách thêm từ “First” hoặc “Top” trước từ “By”.

Ví dụ: findTopBy…, findTop1By…

Spring data JPA còn đưa ra @Query anntation để có thể linh động hơn trong việc tương tác với dữ liệu (data) trong database với JPQL hoặc SQL.

Ví dụ:

* JPQL-@Query(“SELECT myEntity FROM MyEntity myEntity WHERE …”)

List<MyEntity> findMyEntityById()

* SQL-@Query(value=“SELECT myEntity FROM MyEntity myEntity WHERE …”, nativeQuery= true)

List<MyEntity> findMyEntityById()

* 1. **Tổng kết đề tài**

Như vậy, với công nghệ Spring framework đã giúp ta có thể giải quyết được những khó khăn trong việc giao tiếp với nhiều cơ sở dữ liệu(database). Những ứng dụng có thể tái sử dụng với nhiều database, tiết kiệm rất nhiều thời gian và công sức cho lập trình viện và có thể tránh được tình trạng phải ánh xạ cố định cho một database duy nhất.

1. **KẾT QUẢ ĐẠT ĐƯỢC QUA ĐỢT THỰC TẬP**
   1. **Những kiến thức lý thuyết đã được cũng cố**

Qua những tháng khi thực tập tôi đã được củng cô về những lý thuyết đã được học ở trường như :

* Java Core
* Java Collections
* Thread
  1. **Những kỹ năng thực thành đã được học thêm**
     1. **Maven project**

Maven project là một công cụ do Apache tạo ra, dùng để quản lý, thiết lập tự động một dự án phần mềm, chủ yếu dùng cho các lập trình viên java.

Maven hỗ trợ việc tự động hóa các quá trình tạo dự án ban đầu, thực hiện biên dịch, kiểm thử, đóng gói và triển khai sản phẩm.

Được phát triển bằng ngôn ngữ Java cho phép Maven chạy trên nhiều nền tảng khác nhau: Windows, Linux và Mac OS...

Maven dùng khái niệm Project Object Model (POM) để mô tả việc build project, các thành phần phụ thuộc và các module. Nó định nghĩa trước các target cho việc khai báo task, trình biên dịch, đóng gói và thứ tự hoạt động để mọi việc diến ra tốt nhất.

Trong mỗi project Maven tạo ra một file .pom, trong file này định nghĩa ra những task như task khi chạy test, task khi build và khi chạy Maven sẽ dựa vào những định nghĩa này để thao tác với project.

* + 1. **Design Parttern**

Design patterns là các giải pháp đã được tối ưu hóa, được tái sử dụng cho các vấn đề lập trình mà chúng ta gặp phải hàng ngày. Nó là một khuôn mẫu đã được suy nghĩ, giải quyết trong tình huống cụ thể rồi.

Các vấn đề mà bạn gặp phải có thể bạn sẽ tự nghĩ ra cách giải quyết nhưng có thể nó chưa phải là tối ưu. Design Pattern giúp bạn giải quyết vấn đề một cách tối ưu nhất, cung cấp cho các giải pháp trong lập trình.

Nó không phải là ngôn ngữ cụ thể nào cả. Design patterns có thể thực hiện được ở phần lớn các ngôn ngữ lập trình. Ta thường gặp nó nhất trong lập trình OOP.

Có 3 nhóm chính sau:

* **Creational Pattern** (nhóm khởi tạo) gồm: Abstract Factory, Factory Method, Singleton, Builder, Prototype. Nó sẽ giúp bạn trong việc khởi tạo đối tượng, như bạn biết để khởi tạo bạn phải sử dụng từ khóa new, nhóm **Creational Pattern** sẽ sử dụng một số thủ thuật để khởi tạo đối tượng mà bạn sẽ không nhìn thấy từ khóa này.
* **Structural Pattern** (nhóm cấu trúc) gồm: Adapter, Bridge, Composite, Decorator, Facade, Proxy và Flyweight. Nó dùng để thiết lập, định nghĩa quan hệ giữa các đối tượng.
* **Behavioral Pattern** gồm: Interpreter, Template Method, Chain of Responsibility, Command, Iterator, Mediator, Memento, Observer, State, Strategy và Visitor. Nhóm này dùng trong thực hiện các hành vi của đối tượng.
  1. **Những kinh nghiệm đã được tích lũy**

Trải qua hai tháng thực tập tôi thực sự cảm nhận được mục đích của việc thực tập cho sinh viên đi thực tập những kỹ năng chuyên môn nghiệp vụ mà mình học là rất cần thiết. Vì những kiến thức mà tôi học trện lớp mới chỉ là lý thuyết, còn việc thực hành kiến thức đó thì chưa được chú trọng. Cho nên khi đi thực tập , giúp tôi thao tác thực hành và hiểu hơn những môn học mà thầy cô đã tận tình dạy bảo và truyền đạt. Thực tập không chỉ là điều hết sức quan trong đối với sinh viên ngành Công nghệ thông tin mà còn là những nghành khác nữa, đây la 2 điều kiện để chuẩn bị hành trang cho sinh viên sau khi ra trường có thể vận dụng kiến thức mình học được để làm việc.

TÀI LIỆU THAM KHẢO

[ 1 ]<https://docs.spring.io/spring/docs/current/spring-framework-reference/>

(truy cập ngày: 17/11/2017).

[ 2 ]<https://docs.spring.io/spring/docs/current/springframeworkreference/core.html>

(truy cập ngày: 17/11/2017).

[ 3 ]<https://docs.spring.io/spring-boot/docs/1.5.8.RELEASE/reference/htmlsingle/>

(truy cập ngày: 17/11/2017).

[ 4 ]<https://docs.spring.io/spring-data/jpa/docs/current/reference/html/>

(truy cập ngày: 17/11/2017).

[ 5 ]<https://www.tutorialspoint.com/jpa/>

(truy cập ngày: 17/11/2017)

[ 6 ]<http://spsvietnam.vn>

(truy cập ngày: 17/11/2017).