

**TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM  
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG  
KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN GIỮA KÌ**

# **TRANG WEB BÁN HÀNG ONLINE SỬ DỤNG CÔNG NGHỆ SPRING MVC**

*Người hướng dẫn:* **TS NGUYỄN THANH QUÂN**

*Người thực hiện:* **TRẦN TẤN HÙNG - 52000052**

**BÙI QUỐC KHÁNH - 52000771**

**Lớp : TRẦN TẤN HÙNG - 20050201**

**BÙI QUỐC KHÁNH - 20050301**

**Khoá : 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2014**

**TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM  
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG  
KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN GIỮA KÌ**

# **TRANG WEB BÁN HÀNG ONLINE SỬ DỤNG CÔNG NGHỆ SPRING MVC**

*Người hướng dẫn:* **TS NGUYỄN THANH QUÂN**

*Người thực hiện:* **TRẦN TẤN HƯNG - 5200052**

**BÙI QUỐC KHÁNH - 52000771**

**Lớp : TRẦN TẤN HƯNG - 20050201**

**BÙI QUỐC KHÁNH - 20050301**

**Khoá : 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2014**

## **LỜI CẢM ƠN**

Chúng em chân thành cảm ơn thầy Nguyễn Thanh Quân đã dành giờ gian truyền dạy cho chúng em kiến thức về môn lập trình web.

## **ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của TS Nguyễn Văn A;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

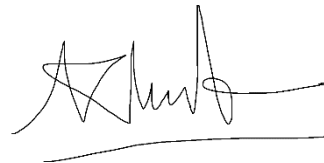
Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*



*Bùi Quốc Khánh*

*Trần Tấn Hưng*

## **PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN**

### **Phần xác nhận của GV hướng dẫn**

---

---

---

---

---

---

---

Tp. Hồ Chí Minh, ngày      tháng      năm  
(kí và ghi họ tên)

### **Phần đánh giá của GV chấm bài**

---

---

---

---

---

---

---

Tp. Hồ Chí Minh, ngày      tháng      năm  
(kí và ghi họ tên)

## TÓM TẮT

Đề tài của nhóm em là trang web bán hàng online sử dụng công nghệ mvc.

Nội dung của đề tài sẽ bao gồm lý thuyết, source code của trang web được viết bằng công nghệ spring mvc, cơ sở dữ liệu sử dụng mysql của Xampp, video giải thích code cũng như những chức năng có trong trang web.

## MỤC LỤC

LỜI CẢM ƠN .....	i
PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN .....	iii
TÓM TẮT .....	iv
MỤC LỤC .....	1
DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ .....	3
CHƯƠNG 1 – KHÁI NIỆM .....	4
CHƯƠNG 2 – CÁC THÀNH PHẦN CỦA MÔ HÌNH MVC TRONG JAVA .....	6
2.1 Model .....	6
2.2 View .....	6
2.3 Controller .....	6
CHƯƠNG 3 – ƯU ĐIỂM CỦA MÔ HÌNH MVC .....	7
CHƯƠNG 4 – MỘT SỐ ANNOTATIONS .....	8
4.1 @Controller: .....	8
4.2 @RequestMapping: .....	8
4.3 @ModelAttribute: .....	9
CHƯƠNG 5 - JPA .....	11
TÀI LIỆU THAM KHẢO .....	13

## **DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT**

### **CÁC CHỮ VIẾT TẮT**

MVC	Model – View – Controller
JPA	Java Persistence API
API	Application Programming Interface
ORM	Object Relational Mapping
SQL	Strutured Query Language
HTTP	Hypertext Transfer Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator



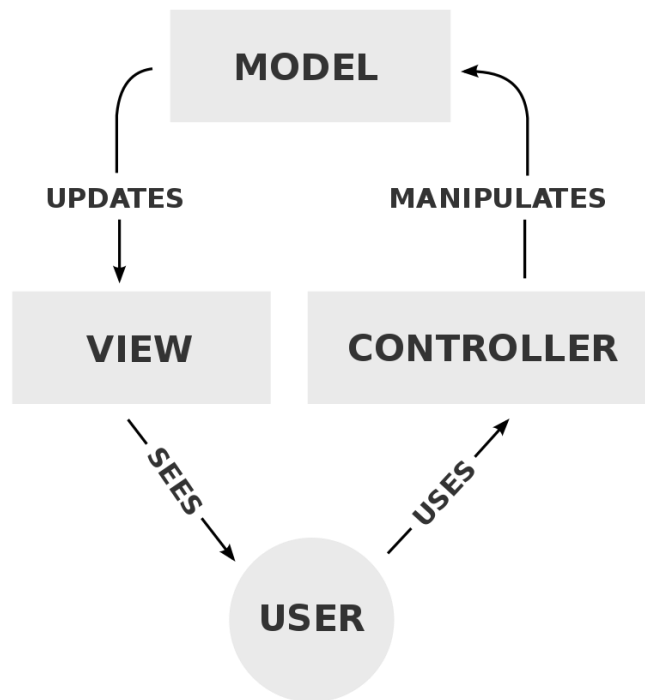
## **DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ**

### **DANH MỤC HÌNH**

Hình 1. 1 : Kiến trúc MVC theo tầng.....	4
Hình 1. 2 : Mô tả luồng đi trong mô hình MVC .....	5
Hình 5. 1 Mô hình JPA .....	11
Hình 5. 2 : Các thành phần của JPA .....	11

## CHƯƠNG 1 – KHÁI NIỆM

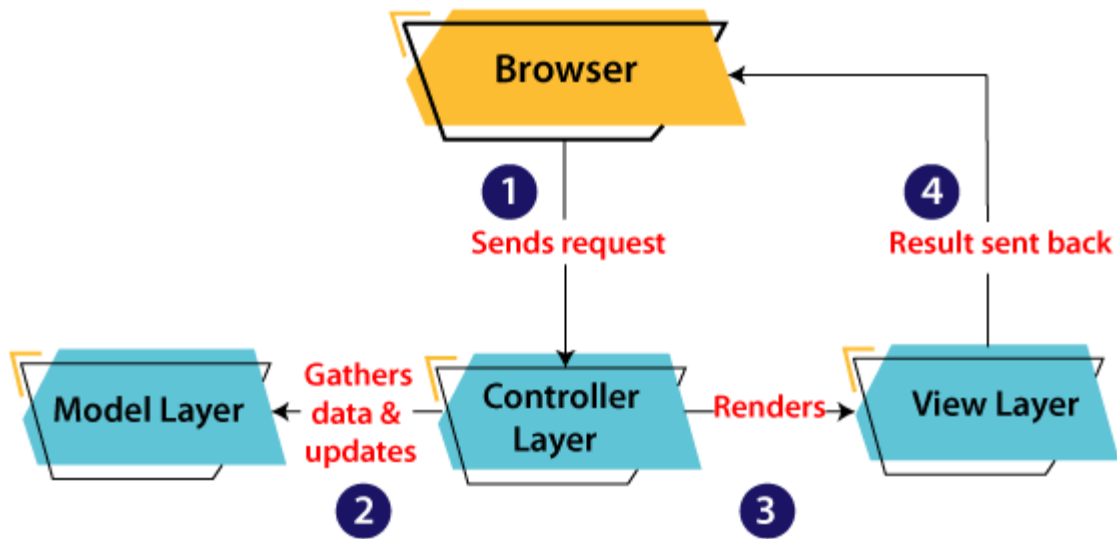
Model – view – controller là một mẫu kiến trúc phần mềm thường được sử dụng để phát triển các giao diện người dùng chia logic chương trình liên quan thành ba phần tử được kết nối với nhau. Điều này được thực hiện để tách biệt các biểu diễn thông tin bên trong khỏi cách thông tin được trình bày và chấp nhận từ người dùng.



Hình 1. 1 : Kiến trúc MVC theo tầng

Kiến trúc MVC bao gồm 3 tầng:

- Model: Có chức năng quản lý và xử lý dữ liệu.
- View: Có nhiệm vụ hiển thị giao diện cho người dùng.
- Controller: có chức năng điều khiển luồng dữ liệu trong ứng dụng cho phép tương tác giữa model và view.



Hình 1. 2 : Mô tả luồng đi trong mô hình MVC

Mô tả luồng đi trong mô hình MVC:

1. Khi client thực hiện request tới server sẽ được chuyển qua controller
2. Controller sẽ gọi đến Model để xử lý hay update các dữ liệu.
3. Sau đó controller sẽ chuyển dữ liệu đã được xử lý và render ra màn hình ở tầng view.

Cuối cùng kết quả sẽ được chuyển về browser cho client.

## **CHƯƠNG 2 – CÁC THÀNH PHẦN CỦA MÔ HÌNH MVC TRONG JAVA**

### **2.1 Model**

Một trong những thành phần quan trọng nhất của mô hình MVC trong Java. Đây là bộ phận chịu trách nhiệm quản lý dữ liệu. Mô hình có chức năng truyền tải thông tin cần hiển thị đến người dùng thông qua màn hình và xử lý thông tin để người dùng dễ dàng tiếp cận.

Model hoàn toàn độc lập với các thành phần MVC còn lại và chứa các tác vụ cần thiết nhất cho quá trình lập trình.

### **2.2 View**

Thành phần tiếp theo trong mô hình MVC trong Java là View. Đối với người dùng, View có một vai trò thiết yếu. View thực hiện nhiệm vụ tạo tương tác người dùng và hiển thị kết quả từ controller. Đồng thời View cũng nhận các hoạt động của người dùng và yêu cầu chuyển cho Controller xử lý.

### **2.3 Controller**

Nói đến các thành phần trong một mô hình MVC nào đó thì không thể bỏ qua bộ điều khiển. Không có thành phần này, tất cả các hoạt động của mô hình hoặc chế độ xem đều không có giá trị.

Controller thực hiện chức năng tương tác giữa View và Model. Định nghĩa lệnh và chạy xử lý lệnh trên hệ thống. Bộ điều khiển thu thập các hành động của người dùng từ Chế độ xem và tương tác với Mô hình để truyền tải thông tin cần thiết cho người dùng.

## CHƯƠNG 3 – ƯU ĐIỂM CỦA MÔ HÌNH MVC

**Vai trò riêng biệt:** mô hình spring mvc tách biệt từng vai trò, trong đó đối tượng mô hình, bộ điều khiển (controller), view, DispatcherServlet,.. được thực hiện bởi một đối tượng chuyên biệt.

**Dung lượng nhẹ:** mô hình sử dụng servlet-container trọng lượng nhẹ để phát triển và triển khai mô hình.

**Cấu hình mạnh:** cung cấp một cấu hình mạnh mẽ cho lớp khung và lớp ứng dụng.

**Dễ dàng kiểm thử:** trong spring, thường thì ta có thể tạo các Javabean cho phép chèn dữ liệu kiểm tra bằng phương thức setter.

**Tái sử dụng code:** thay vì tạo các đối tượng mới, mô hình cho phép sử dụng các đối tượng hiện có.

**Ánh xạ linh hoạt:** cung cấp các annotation cụ thể để dễ dàng chuyển hướng trang.

## CHƯƠNG 4 – MỘT SỐ ANNOTATIONS

### 4.1 @Controller:

@Controller là một cấp lớp ( class - level) annotation. Nó đánh dấu một lớp là một trình xử lý yêu cầu của web, thường được sử dụng để phục vụ các trang web. Mặc định, nó sẽ trả về một chuỗi cho biết đường truyền nào được chuyển hướng. Chủ yếu thường được sử dụng với @RequestMapping annotation.

Annotation này hoạt động như một khuôn mẫu cho lớp được chú thích, khi dùng annotation @Controller thì spring container sẽ nhận biết được lớp được khai báo phía dưới là một controller.

### 4.2 @RequestMapping:

@RequestMapping là một trong những annotation phổ biến nhất được sử dụng trong mô hình spring, được sử dụng để ánh xạ các yêu cầu web tới các phương thức xử lý của MVC và Rest controller.

Annotation @RequestMapping được áp dụng cho cấp độ lớp hoặc cấp độ phương thức trong controller. @RequestMapping cấp lớp ánh xạ một đường dẫn hoặc mẫu request cụ thể lên controller. Tiếp theo, ta có thể sử dụng @RequestMapping cấp phương thức bổ sung để làm ánh xạ cụ thể hơn.

- @RequestMapping với HTTP Method: Annotation, @RequestMapping có khả năng xử lý các phương thức yêu cầu HTTP, chẳng hạn như GET, PUT, POST, DELETE và PATCH. Ví dụ

```
@RequestMapping(method = RequestMethod.GET)
```

- @RequestMapping với nhiều URI : có thể có nhiều ánh xạ yêu cầu cho một phương thức. Ví dụ:

```
@RequestMapping(value={"", "/page", "page*", "view/*"})
```

- **@RequestMapping** với Request Parameters: sử dụng phần tử params, có thể có nhiều phương thức xử lý yêu cầu xử lý các yêu cầu cho cùng một URL, nhưng với các tham số khác nhau. Ví dụ:

```
@RequestMapping(value="/fetch",params={"personId=10"})
```

Mapping Annotation phân biệt Method: Các annotation này thể hiện tốt hơn các ngữ nghĩa của các phương thức xử lý. hoạt động như một trình bao bọc cho @RequestMapping.

- **@GetMapping:** là một annotation tổng hợp hoạt động như một lối tắt cho @RequestMapping (method = RequestMethod.GET)
- **@PostMapping:** là một annotation tổng hợp hoạt động như một lối tắt cho @RequestMapping (method = RequestMethod.POST)
- **Ngoài ra còn có:** @PutMapping, @DeleteMapping, @PatchMapping.

#### 4.3 @ModelAttribute:

Annotation @ModelAttribute là một annotation liên kết một tham số phương thức hoặc giá trị trả về của phương thức với một thuộc tính mô hình được đặt tên, sau đó hiển thị nó trong chế độ xem web.

- Ở cấp method: khi chúng ta sử dụng chú thích ở cấp phương thức, nó cho biết mục đích của phương thức là thêm một hoặc nhiều thuộc tính mô hình.  

```
model.addAttribute("msg", "Welcome to Bangkok!");
```

Trong ví dụ trên, chúng ta thấy một phương thức bổ sung một thuộc tính có tên msg cho tất cả các mô hình được định nghĩa trong lớp trình điều khiển.

- Là đối số cho một phương thức: khi chúng ta sử dụng chú thích làm đối số phương thức, nó biểu thị để truy xuất đối số từ mô hình. Khi chú thích không xuất hiện, trước tiên chú thích đó phải được khởi tạo và sau đó thêm

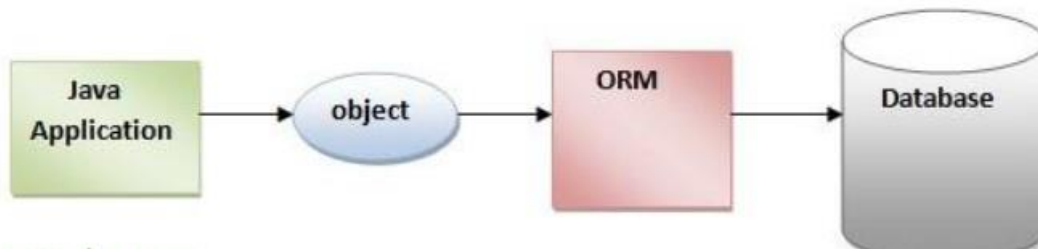
vào mô hình. Sau khi xuất hiện trong mô hình, các trường đối số sẽ được điền từ tất cả các tham số yêu cầu có tên phù hợp.

```
public String submit(@ModelAttribute("employee") Employee  
employee) {}
```



## CHƯƠNG 5 - JPA

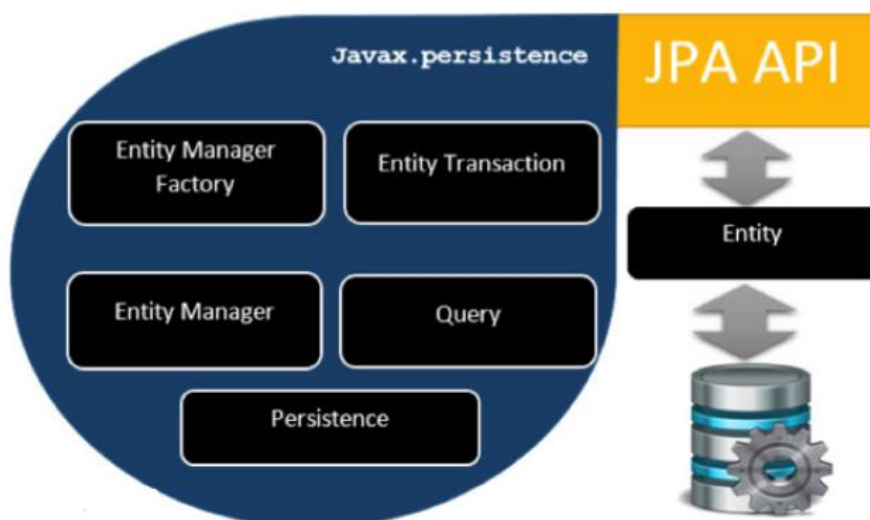
**JPA (Java Persistence API)** , nó là một đặc tả Java cho việc ánh xạ giữa các đối tượng Java với cơ sở dữ liệu quan hệ sử dụng công nghệ phổ biến là **ORM** (Object Relational Mapping).



Hình 5. 1 Mô hình JPA

ORM là một công nghệ/ khái niệm/ quá trình chuyển đổi dữ liệu từ ngôn ngữ hướng đối tượng sang Database quan hệ và ngược lại.

JPA sử dụng metadata để ánh xạ các đối tượng persistence với các bảng trong cơ sở dữ liệu. JPA hỗ trợ SQL như là một ngôn ngữ truy vấn để dễ dàng xử lý các truy vấn cơ sở dữ liệu. Ngôn ngữ truy vấn JPA có thể dùng thực thi cả truy vấn tĩnh và truy vấn động.



Hình 5. 2 : Các thành phần của JPA

Ba thành phần chính là: **Entity**, **EntityManager** và **EntityManagerFactory**. Ngoài ra còn có, EntityTransaction, Persistence, Query.

- **Entity:** Entity là các đối tượng thể hiện tương ứng 1 table trong cơ sở dữ liệu. Entity thường là các class POJO đơn giản, chỉ gồm các phương thức getter, setter.
  - + Đây là những thuộc tính của một thực thể mà một đối tượng phải có: **Persistability, Persistent Identity, Transactionality, Granuality** .
  - + Để chuyển đối lớp thành một thực thể, hãy thêm chú thích **@Entity** và **@Id** vào trong đó.
- **EntityManager:** EntityManager là một interface cung cấp các API cho việc tương tác với các Entity.
  - + Trình quản lý thực thể triển khai API và gói gọn tất cả chúng trong một giao diện duy nhất.
  - + Được sử dụng để đọc, xóa và ghi một thực thể.
  - + Một đối tượng được tham chiếu bởi một thực thể được quản lý bởi trình quản lý thực thể.
- **EntityManagerFactory:** EntityManagerFactory được dùng để tạo ra một instance của EntityManager.

## TÀI LIỆU THAM KHẢO

### Tiếng Việt

1. Nguyễn Hưng (2022), “Tìm hiểu mô hình MVC là gì? Ví dụ về cách sử dụng mô hình MVC”, [MVC là gì? Ứng dụng của mô hình MVC trong lập trình \(vietnix.vn\)](#)
2. mona.media , “Mô hình MVC là gì và ứng dụng của MVC trong lập trình” [Mô hình MVC là gì và ứng dụng của MVC trong lập trình \(mona.media\)](#)
3. Giang Phan (2019), “Tổng quan về JPA (Java Persistence API)”, <https://topdev.vn/blog/tong-quan-ve-jpa-java-persistence-api/>
4. Nguyễn Khách Tùng (2020), “Sử dụng Annotation @RequestMapping trong Spring MVC”, <https://codegym.vn/blog/2020/06/02/su-dung-annotation-requestmapping-trong-spring-mvc/#>
5. Tống Hoàng Vũ (2021), “Spring Boot xử lý request trong Controller như thế nào (phần 1)”, <https://viblo.asia/p/spring-boot-xu-ly-request-trong-controller-nhu-the-nao-phan-1-gGJ59ANj5X2>

### Tiếng Anh

6. javaTpoint , “Spring MVC Tutorial”, [Spring MVC Tutorial - javatpoint](#)
7. javaTpoint , “Spring MVC Form Example”, [Spring MVC Form Example - javatpoint](#)
8. javaTpoint , “Spring vs. Spring Boot vs. Spring MVC”, [Spring vs Spring Boot vs Spring MVC - javatpoint](#)
9. javaTpoint , “Spring Boot annotations”, <https://www.javatpoint.com/spring-boot-annotations>
10. spring.io, “Spring Data JPA” , <https://spring.io/projects/spring-data-jpa>