**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**



**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**--------------------**

**ĐỒ ÁN MÔN MỘT SỐ THUẬT TOÁN THÔNG MINH**

**Đề tài: GAME CỜ CARO SỬ DỤNG AI TRÊN C#**

**Giảng viên hướng dẫn:**

**Ths. NGUYỄN TRÁC THỨC**

**Sinh viên thực hiện:**

**NGUYỄN VĂN NGUYỆN – 13520567**

**HUỲNH NGỌC THẮNG – 13520775**

**TRẦN MINH TUẤN -**

***Năm học 2007 - 2008***

***TPHCM, 31/05/2017***

**Mục lục**

[PHẦN MỘT: GIỚI THIỆU 3](#_Toc483926414)

[1. TỔNG QUAN VỀ J2EE 3](#_Toc483926415)

[1.1.1 JavaEE là gì? 3](#_Toc483926416)

[1.1.2 Kiến trúc của JavaEE 3](#_Toc483926417)

[1.1.3 Các Container trong JavaEE 5](#_Toc483926418)

[1.1.4 Các công nghệ trong JavaEE 5](#_Toc483926419)

[1.1.5 Nhận xét về JavaEE 6](#_Toc483926420)

[2. TỔNG QUAN VỀ PROJECT 6](#_Toc483926421)

[PHẦN HAI: CÔNG NGHỆ ÁP DỤNG 6](#_Toc483926422)

[1. TỔNG QUAN VỀ MVC 6](#_Toc483926423)

[1.1.1 Mô hình MVC là gì? 6](#_Toc483926424)

[1.1.2 Các thành phần trong MVC 6](#_Toc483926425)

[1.1.3 Sự tương tác giữa các lớp trong mô hình MVC 7](#_Toc483926426)

[1.1.4 Quy trình hoạt động mô hình MVC trong dự án Website 7](#_Toc483926427)

[1.1.5 Ưu nhược điểm của mô hình MVC? 8](#_Toc483926428)

[2. TỔNG QUAN VỀ STRUTS FRAMEWORK 8](#_Toc483926429)

[2.1 Struts Framework là gì? 8](#_Toc483926430)

[2.2 Các kiến trúc lập trình 9](#_Toc483926431)

[2.3 Các thành phần cơ bản của Struts 9](#_Toc483926432)

[2.4 Chu kì sống của Struts 10](#_Toc483926433)

[PHẦN BA: THIẾT KẾ VÀ CÀI ĐẶT 10](#_Toc483926434)

[1. KIẾN TRÚC HỆ THỐNG 10](#_Toc483926435)

[2. SƠ ĐỒ USE – CASE 10](#_Toc483926436)

[3. SƠ ĐỒ SEQUENCE 10](#_Toc483926437)

[4. GIAO DIỆN ỨNG DỤNG 10](#_Toc483926438)

[NHÂN XÉT VÀ HƯỚNG PHÁT TRIỂN 10](#_Toc483926439)

[TÀI LIỆU THAM KHẢO 10](#_Toc483926440)

# **PHẦN MỘT: GIỚI THIỆU**

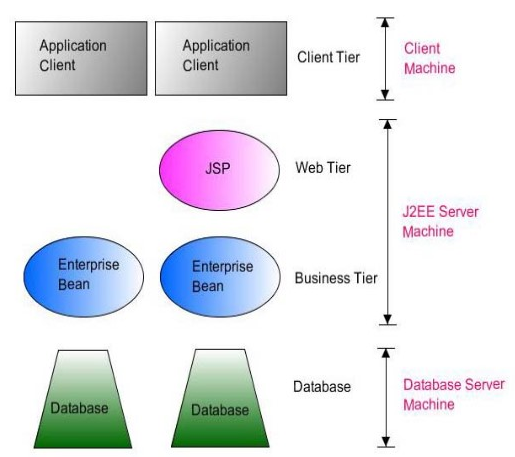
## **TỔNG QUAN VỀ J2EE**

### JavaEE là gì?

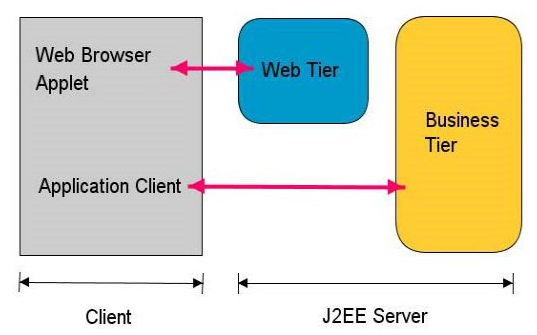
Java 2 Platform Enterprise Edition viết tắt là J2EE, định nghĩa một chuẩn để phát triển những ứng dụngthương mại đa tầng (multitier enterprise applications). J2EE platform chứa đựng những tính năng tốt của Java 2 Platform như “viết một lần, chạy ở mọi nơi”, JDBC (Java Database Connectivity) truy cập cơ sở dữ liệu, CORBA kết hợp với những tài nguyên đã có…Được xây dựng trên nền tảng là Java 2 Platform, J2EE mở rộng hổ trợ cho nhiều công nghệ mới như Enterprise JavaBeans, Java Servlets, JavaServer Pages và XML.

### Kiến trúc của JavaEE

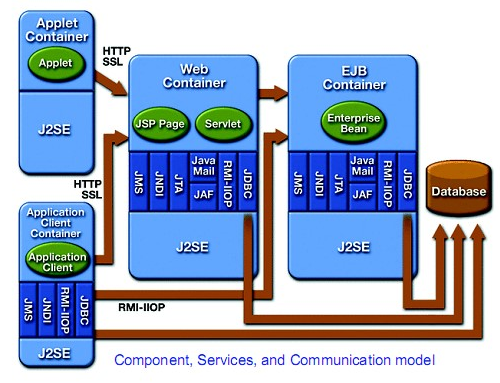
* J2EE nền tảng sử dụng một mô hình ứng dụng phân tán đa tầng.



* Trong mô hình ứng dụng J2EE có nhiều tầng:
* Tầng khách hàng (client tier), tầng web (web tier),Tầng thương mại(business tier)và tầng hệ thống thông tin thương mại (enterprise information system tier).
* Tầng thương mại và tầng web nằm trên một máy chủ ứng dụng gọi là máy chủ ứng dụng (application server) hay máy chủ J2EE (J2EE server)
* .Máy chủ J2EE cung cấp những dịch vụ cần thiết cho những thành phần (component) của tầng thương mại và tầng web.



* Có hai loại khách hàng (client) là: Những khách hàng ứng dụng (application clients) và những khách hàng web (web client).
* Khách hàng web truy cập tới những thành phần trong tâng web là các lớp Java Servlet hay Java ServerPages.
* Những khách hàng ứng dụng là những ứng dụng độc lập, chúng truy cập trực tiếp tới các thành phần trong tầng thương mại



### Các Container trong JavaEE

* **Web Container** : Quản lý sự thực thi của các thành phần web, và container của chúng chạy trong máy chủ J2EE (J2EE server).
* **EJB Container** : Quản lý sự thực thi của các thành phần EJB, các thành phần EJB và container của chúng chạy trong J2EE server.
* **Application client Container** : Quản lý sự thực thi các thành phần của ứng dụng client.
* **Applet Container** : Quản lý sự thực thi của applets(ứng dụng ký sinh).

### Các công nghệ trong JavaEE

* **Java Database Connectivity (JDBC) API**: JDBC là một tập hợp các giao diện cho phép các ứng dụng Java truy cập vào cơ sở dữ liệu bất kỳ. API này có cùng một mục đích như ODBC (Open Database Connectivity) Microsoft.
* **Remote Method Invocation (RMI)**: RMI là một API cho phép các đối tượng Java để giao tiếp từ xa với các đối tượng khác.
* **Java IDL**: IDL (Interface Definition Language) là một ngôn ngữ nền tảng tiêu chuẩn khai báo độc lập được sử dụng để xác định các giao diện đối tượng triển khai cung cấp và đối tượng khách hàng gọi. Java IDL cho phép bất kỳ đối tượng Java để giao tiếp với các đối tượng khác trong ngôn ngữ bất kỳ bằng phương tiện của IDL.
* **Enterprise Java Beans**: Đóng gói các logic thương mại (business logic) và hổ trợ thực hiện giao tác và bảo mật.
* **Servlets và Java Server Pages (JSP):** Công nghệ dùng trong xây dựng những ứng dụng web.
* **Java Message Service (JMS)**: Tập APIs cho phép ứng dụng tạo, gởi, nhận, và đọc thông điệp bất đồng bộ.
* **Java Transaction API (JTA):** JTA cho phép những ứng dụng J2EE thực hiện những giao tác phân tán (distibuted transaction).
* **JavaMail:** Cho phép thành phần java gởi và nhận mail.
* **Java API cho xử lý XML (Java API for XML Processing -JAXP**): XML là một định dạng dữ liệu cho các tài liệu thay thế cấu trúc trên Web. JAXP cho phép các ứng dụng Java phân tích và chuyển đổi tài liệu XML. XML được sử dụng nhiều trong J2EE như là một định dạng dữ liệu.
* **Java Naming và Directory Interface (JNDI)**: JNDI là một giao thức cung cấp một tập API chuẩn cho phép truy cập tới những dịch vụ naming và directory.
* **Java Persistence API (JPA)** : Cho phép ánh xạ giữa một lớp java với các cột trong một bảng.

### Nhận xét về JavaEE

## **TỔNG QUAN VỀ PROJECT**

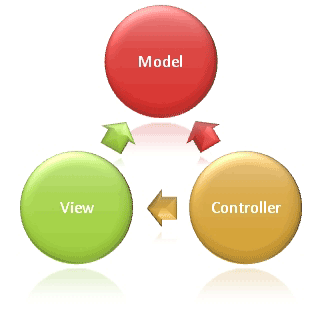
# **PHẦN HAI: CÔNG NGHỆ ÁP DỤNG**

## **TỔNG QUAN VỀ MVC**

### Mô hình MVC là gì?

Mô hình MVC là một chuẩn mô hình và đóng vai trò quan trọng trong quá trình xây dựng – phát triển – vận hành và bảo trì một hệ thống hay một ứng dụng – phần mềm. Nó tạo ra một mô hình 3 lớp Model – View – Controller tách biệt và tương tác nhau, giúp các chuyên gia có thể dễ dàng dựa vào mô hình để trao đổi và xử lý những nghiệp vụ một cách nhanh chóng. Đây là một mô hình đã xuất hiện từ những năm 70 của thế kỷ 20 tại phòng thí nghiệm Xerox PARC ở Palo Alto, nó không phụ thuộc vào môi trường, nền tảng xây dựng hay ngôn ngữ phát triển. Chúng ta có thể áp dụng mô hình MVC vào các dự án trong môi trường Windows, Linux… và sử dụng bất kỳ ngôn ngữ nào như PHP, ASP, JSP…

### Các thành phần trong MVC



Mô hình MVC được chia làm 3 lớp xử lý gồm Model – View – Controller :

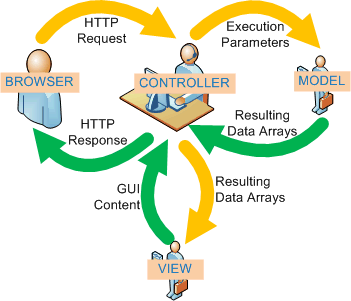
* **Model** : là nơi chứa những nghiệp vụ tương tác với dữ liệu hoặc hệ quản trị cơ sở dữ liệu (mysql, mssql… ); nó sẽ bao gồm các class/function xử lý nhiều nghiệp vụ như kết nối database, truy vấn dữ liệu, thêm – xóa – sửa dữ liệu…
* **View** : là nới chứa những giao diện như một nút bấm, khung nhập, menu, hình ảnh… nó đảm nhiệm nhiệm vụ hiển thị dữ liệu và giúp người dùng tương tác với hệ thống.
* **Controller** : là nới tiếp nhận những yêu cầu xử lý được gửi từ người dùng, nó sẽ gồm những class/ function xử lý nhiều nghiệp vụ logic giúp lấy đúng dữ liệu thông tin cần thiết nhờ các nghiệp vụ lớp Model cung cấp và hiển thị dữ liệu đó ra cho người dùng nhờ lớp View

### Sự tương tác giữa các lớp trong mô hình MVC

* **Controller – View** sẽ lấy những hình ảnh, nút bấm…hoặc hiển thị dữ liệu được trả ra từ Controller để người dùng có thể quan sát và thao tác. Trong sự tương tác này cũng có thể không có dữ liệu được lấy từ Model và khi đó nó chỉ chịu trách nhiệm hiển thị đơn thuần như hình ảnh, nút bấm…
* **Controller – Model** là luồng xử lý khi controller tiếp nhận yêu cầu và các tham số đầu vào từ người dùng, controller sẽ sử dụng các lớp/hàm trong Model cần thiết để lấy ra những dữ liệu chính xác.
* **View – Model** có thể tương tác với nhau mà không qua Controller, nó chỉ đảm nhận hiển thị dữ liệu chứ không phải qua bất kỳ xử lý nghiệp vụ logics nào. Nó giống như các vùng dữ liệu hiển thị tĩnh trên các website như block slidebar…

Mô hình MVC có thể áp dụng cho nhiều loại dự án khác nhau : dự án website, dự án ứng dụng – phần mềm,… nhưng trong phạm vi bài viết chúng ta sẽ chỉ đi tìm hiểu mô hình này được áp dụng như nào trong các dự án xây dựng và phát triển website

### Quy trình hoạt động mô hình MVC trong dự án Website



Mô hình trên là thể hiện mô hình MVC trong các dự án website và sẽ hoạt động theo quy trình sau:

* Người dùng sử dụng một BROWSER trình duyệt web bất kỳ (Firefox, Chrome, IE,…) để có thể gửi những yêu cầu (HTTP Request) có thể kèm theo những dữ liệu nhập tới những CONTROLLER xử lý tương ứng. Việc xác định Controllerr xử lý sẽ dựa vào một bộ Routing điều hướng.
* Khi CONTROLLER nhận được yêu cầu gửi tới, nó sẽ chịu trách nhiệm kiểm tra yêu cầu đó có cần dữ liệu từ MODEL hay không? Nếu có, nó sẽ sử dụng các class/function cần thiết trong MODEL và nó sẽ trả ra kết quả( Resulting Arrays), khi đó CONTROLLER sẽ xử lý giá trị đó và trả ra VIEW để hiển thị. CONTROLLER sẽ xác định các VIEW tương ứng để hiển thị đúng với yêu cầu.
* Khi nhận được dữ liệu từ CONTROLLER, VIEW sẽ chịu trách nhiệm xây dựng các thành phẩn hiển thị như hình ảnh, thông tin dữ liệu… và trả về GUI Content để CONTROLLER đưa ra kết quả lên màn hình BROWSER.
* BROWSER sẽ nhận giá trị trả về( HTTP Response) và sẽ hiển thị với người dùng. Kết thúc một quy trình hoạt động.

### Ưu nhược điểm của mô hình MVC?

#### Ưu điểm

* Các dự án có thể áp dụng ngay mô hình MVC mà không phụ thuộc môi trường, nền tảng xây dựng hay ngôn ngữ lập trình phát triển;
* Quy hoạch các class/ function vào các thành phần riêng biệt Controller – Model – View, khi đó sẽ dễ dàng xây dựng – phát triển – quản lý – vận hành và bảo trì một dự án, tạo sự rõ ràng, trong sáng trong quá trình phát triển dự án, kiểm soát được các luồng xử lý và tạo ra các thành phần xử lý nghiệp vụ chuyên biệt hóa.
* Tạo thành mô hình chuẩn cho nhiều dự án, các chuyên gia sẽ tiếp cận – tìm hiểu những dự án đó một cách nhanh chóng và hiệu quả . Nếu bạn nắm rõ mô hình MVC của một dự án nào đó, thì khi tiếp cận với một dự án khác mà bạn chưa từng biết hoặc tiếp xúc, nhưng nó lại được xây dựng với mô hình MVC thì sẽ không khó khăn gì mà cực kỳ dễ dàng. Học một nhưng có thể hiểu và sử dụng được mười.
* Giúp các chuyên gia lập trình, nhà quản lý, nhà đầu tư, PM… có thể hiểu được dự án hoạt động ra sao hoặc giúp các lập trình viên dễ dàng quản lý – phát triển dự án. Nó không phải ngôn ngữ, nhưng khi họ cùng nhìn vào nó thì sẽ tự hiểu nó là gì, khi đó họ có thể trao đổi các yêu cầu và bàn bạc công việc.
* Đây là một mô hình chuẩn, nó tối ưu nhất hiện nay so với nhiều mô hình khác và được sử dụng trong nhiều dự án và nhiều lĩnh vực, đặc biệt trong công nghệ sản xuất ứng dụng – phần mềm. Các lập trình viên sử dụng mô hình chuẩn MVC để có thể dễ dàng phân phối và chuyển giao công nghệ.
* Đây là mô hình đơn giản, xử lý những nghiệp vụ đơn giản, và dễ dàng triển khai với các dự án nhỏ.

#### Nhược điểm

Yêu cầu về chuyên môn khá cao, có kiến thức vững về các mô hình chuẩn;Khó triển khai với những dự án yêu cầu phúc tạp hơn. Hiện nay đang có một khái niệm mô hình mới đó là HMVC đang dần thay thế cho MVC

## **TỔNG QUAN VỀ STRUTS FRAMEWORK**

### 2.1 Struts Framework là gì?

Struts là một framework phục vụ việc phát triển các ứng dụng Web trên Java. Sử dụng mẫu thiết kế Model-View-Controller (MVC), Struts giải quyết rất nhiều các vấn đề liên quan đến các ứng dụng Web hướng business đòi hỏi hiệu năng cao sử dụng Java servlet và JSP. Struts cơ bản định hình lại cách các Web programmer nghĩ về và cấu trúc một ứng dụng Web.

Structs được phát triển bởi Craig McClanahan và được bão trợ bởi Apache, nhóm Jakarta

### 2.2 Các kiến trúc lập trình

#### 2.2.1. Kiến trúc truyền thống (Còn gọi là mô hình 1 - Model 1).

Kiến trúc Model 1 hết sức đơn giản. Tóm lại là tất cả mọi thứ đều được gói gọn trong Servlet hoặc JSP từ việc xử lý request, xác nhận tính hợp lệ của dữ liệu, điều quản business logic và generate response. Mặc dù về khái niệm hết sức đơn giản, kiến trúc này không phù hợp cho việc phát triển các ứng dụng quy mô lớn vì chắc chắn một số lượng lớn các tính năng sẽ bị lặp lại trong các trang JSP (lặp lại code).

Cũng vậy, kiến trúc Model 1 tạo nên sự ràng buộc không cần thiết giữa business logic và presentation logic của ứng dụng. Ví dụ, ngoài giao diện HTML, bạn có thể muốn thêm một giao diện WML cho truy nhập wireless. Trong trường hợp này, việc sử dụng Model 1 yêu cầu sự lặp lại không cần thiết của business logic trong các trang sử dụng HTML và WML.

#### 2.2.2. Kiến trúc MVC (Còn gọi là mô hình 2 - Model 2)

Model 2, hay cách gọi thông dụng ngày nay, Model-View-Controller (MVC), giải quyết nhiều vấn đề của Model 1 với việc cung cấp một sự phân tách rõ ràng trong ứng dụng. Trong kiến trúc MVC, một Servlet trung tâm, được gọi là Controller, tiếp nhận tất cả các request cho ứng dụng. Sau đó Controller sẽ xử lý request và làm việc với Model để chuẩn bị những dữ liệu cần thiết cho View (thường là JSP) và forward dữ liệu tới trang JSP. Sau đó JSP sử dụng các dữ liệu đã được chuẩn bị bởi Controller để generate một response về cho browser.

Trong kiến trúc này, business logic và presentation logic được phân tách nhau. Việc tách business code và presentation code giúp ta có thể sử dụng nhiều giao diện cho ứng dụng, chúng có thể là Web, wireless hay GUI. Thêm vào đó, việc phân tách này còn cung cấp việc tái sử dụng lại code một cách hòan hảo.

Có một điều khá lý thú ở đây là tôi đã từng cho rằng việc trong một ứng dụng Web có thể có nhiều Controller thì ứng dụng đó cũng sử dụng MVC. Có nghĩa là cứ một view component ta sẽ có một Controller cho nó. Tuy nhiên điều này là hòan tòan sai với mẫu thiết kế MVC.

### 2.3 Các thành phần cơ bản của Struts

#### 2.3.1. Framework cơ sở

Framework cơ sở cung cấp các tính năng MVC cốt lõi. Nền tảng của framework cơ sở là Controller servlet: ActionServlet. Phần còn lại của framework cơ sở bao gồm các lớp cơ sở mà ứng dụng của bạn sẽ extend và các class tiện ích. Nổi bật nhát trong các lớp cơ sở là lớp Action và lớp ActionForm. Lớp Action được sử dụng bởi ActionServlet để sử lý các request xác định. Lớp ActionForm được sử dụng để capture dữ liệu từ các HTML form và được sử dụng để chuyển dữ liệu trở lại View để generate response.

#### 2.3.2. Thư viện thẻ JSP

Struts đưa ra các thư viện thẻ để hỗ trợ việc lập trình View logic trong JSP. Các thư viện thẻ JSP cho phép các JSP author sử dụng các thẻ giống HTML.

* HTML: Sử dụng để generate các THML form tương tac với Struts API
* Bean: Sử dụng để làm việc với các Java bean object trong JSP
* Logic: Sử dụng để implement các logic điều kiện đơn giản trong JSP
* Nested: Sử dụng để cho phép các mức nested tùy ý

#### 2.3.3. Tiles plugin

Tiles là một JSP template framework giúp thuận tiện cho việc tái sử dụng mã HTML.

#### 2.3.4. Validator plugin

Validator là một framework với khả năng thực hiện xác nhận tính hợp lệ của dữ liệu ở cả phía server và client.

### 2.4 Chu kì sống của Struts

Trước khi đi vào chi tiết các lớp, nghía qua một chút về tiến trình thực hiện của một ứng dụng Web sử dụng Struts cũng có lẽ không thừa.

* Trình duyệt tạo một request tới ứng dụng Struts sẽ được xử lý bới ActionServlet (Controller).
* ActionServlet (Controller) populate subclass của ActionForm (View), do ta viết, với dữ liệu form HTML và invoke method validate() của nó.
* ActionServlet (Controller) execute subclass của Action do ta viết (Controller).
* Subclass của Action (Controller) đó sẽ tương tác với model component và chuẩn bị dữ liệu cho để hiển thị.
* Sau đó subclass của Action (Controller) đó sẽ chuyển điều khiển tới JSP (View).
* JSP (View) sử dụng dữ liệu nhận được để generate response về cho trình duyệt.

# **PHẦN BA: THIẾT KẾ VÀ CÀI ĐẶT**

## **KIẾN TRÚC HỆ THỐNG**

## **SƠ ĐỒ USE – CASE**

## **SƠ ĐỒ SEQUENCE**

## **GIAO DIỆN ỨNG DỤNG**

# **NHÂN XÉT VÀ KẾT LUẬN**

* Nhóm em phần nào cũng đã hoàn thiện và đáp ứng yêu cầu của đồ án
* Có một số khó khăn trong việc lập trình cũng như viết báo cáo (Báo cáo chưa hoàn thiện cho lắm và có một số chỗ sai sót)
* Bên cạnh đó, còn một số vấn đề cần được khắc phục như:
  + Các lỗi tiềm ẩn trong phần mềm
  + Giao diện chương trình chưa đẹp cho lắm
  + Phần mềm chỉ chạy trên các máy tính có cài đặt đầy đủ chương trình hỗ trợ nên khó phát triển rộng rãi

# **HƯỚNG PHÁT TRIỂN**

* Sửa các lỗi hiện có
* Phát triển thêm việc kết nối giữa Client và Server (dưới sự hỗ trợ của Window Server) để triển khai ứng dụng rộng rãi dễ dàng hơn.

# **PHỤ LỤC**

* Bảng phân công công việc

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ tên** | **MSSV** | **Mức độ hoàn thành** | **Công việc được giao** |
| **1** | Nguyễn Văn Nguyện | 13520567 |  |  |
| **2** | Huỳnh Ngọc Thắng | 13520775 |  |  |
| **3** | Trần Minh Tuấn | 13520567 |  |  |

Nhóm xin chân thành gửi lời cảm ơn đến Giảng Viên đã dạy và hướng dẫn chúng em trong thời gian vừa qua để hoàn thành tốt đồ án này một cách tốt nhất và hoàn thiện nhất,xin chân thành cảm ơn!

# **TÀI LIỆU THAM KHẢO**

[1]. https://struts.apache.org/

[2]. https://vi.wikipedia.org/wiki/Struts

[3]. Struts 2 Design and Programming Second Edition by Kudi Kurniawan

[4] Google, YouTube

[5] Một số nguồn tài liệu khác