JSF là một framework phát triển các ứng dụng Web viết bằng Java nhằm làm đơn giản hóa quá trình phát triển giao diện người dùng cho các ứng dụng J2EE. Để tạo ra giao diện hiện thị, JSF dùng dạng cấu trúc cây của các thẻ, mỗi thẻ là một thành phần giao diện (component) và FacesServlet servlet sẽ thực hiện các công đoạn chuyển đổi ra giao diện ứng dụng, cho người dùng với định dạng HTML. JSF bao gồm:

1. Một tập các API để biểu diễn các thành phần UI (giao diện người dùng) và quản lý trạng thái của chúng, xử lý các sự kiện và kiểm tra dữ liệu đầu vào, quy định việc di chuyển trang (page navigation), hỗ trợ tính đa ngôn ngữ và hỗ trợ cho người sử dụng (accessibility).
2. Một số thành phần (Component) có sắn.
3. Hai thư viện chứa thẻ tùy biến JSP (JavaServer Pages custom tag) để có thể biểu diễn một JavaServer Faces interface bên trong một trang JSP.
4. Mô hình sự kiện phía máy chủ (server-side event model).
5. Khả năng quản lý trạng thái.
6. Managed Beans (JavaBeans được tạo bằng dependency injection).

Why JSF?

1. Tạo ra một framework gồm các thành phần giao diện người dùng chuẩn (standard GUI component framework) nhằm giúp cho các công cụ phát triển dễ dàng hơn cho người dùng trong việc tạo GUI chất lượng đồng thời quản lý các kết quả của GUI với xử lý thực thi của chương trình.
2. Định ra một tập các lớp cơ sở của Java (lightweight Java base classes) biểu diễn các thành phần UI, trạng thái mỗi thành phần, và các sự kiện đầu vào. Những lớp này sẽ liên quan đến chu kì sống của GUI, đặc biệt là quản lý trạng thái trong suốt chu trình sống của một trang thành phần GUI đó.
3. Cung cấp một tập các thành phần GUI chung, bao gồm các thành phần HTML form input. Những thành phần này sẽ được dẫn xuất từ tập đơn giản các lớp cơ sở (để cập ở #1) để từ đó có thể định ra các thành phần mới.
4. Cung cấp một mô hình JavaBeans để có thể truyền đi (dispatch) các sự kiện từ các GUI controls phía máy khách đến các xử lý hiện thực cụ thể từ phía ứng dụng máy chủ.
5. Định ra các APIs để kiểm chứng dữ liệu nhập, bao gồm hỗ trợ kiểm chứng từ phía máy chủ.
6. Chỉ định một mô hình để có thể đa ngôn ngữ hóa hay địa phương hóa các GUI.
7. Khởi tạo tự động dữ liệu ra phù hợp cho các máy khách đích, dựa vào mọi dữ liệu cấu hình ở máy khách đó, bao gồm cả dựa vào phiên bản trình duyệt.
8. Việc khởi tạo tự động dữ liệu ra kèm theo các đòi hỏi về hỗ trợ người dùng (accessibility), được quy định bởi WAI.

Why JSF ?

1. JSF framework được xây dựng trên nền tảng MVC, nhưng được đưa thêm khả năng hỗ trợ event.
2. JSF còn hỗ trợ cơ chế rendering trong việc hỗ trợ giao diện cho người dùng trong quá trình xử lý. Đây là đặc tính nỗi bật của JSF. Hỗ trợ tính toán, xử lý kết quả để kết xuất trình bày giá trị trong các thành phần UI để trình bày giao diện chính xác và hợp lý từ kết xuất của chương trình trong lúc thực thi.
3. JSF có một tập các API biểu diễn các thành phần UI và quản lý các trạng thái của chúng – validation, conversion, multi language.
4. Managed Bean
   1. Bản chất là Java Beans được dùng kết hợp với các thành phần UI interface – các phương thức hàm hỗ trợ trình bày giao diện cho người dùng, trao đổi dữ liệu với các thành phần khác trong ứng dụng, lưu trữ và chuyển xử lý khi người dùng tương tác
   2. Hỗ trợ 3 tầm vực khác nhau trong ứng dụng khi thao tác như request, session và application. Tầm vực none – chỉ là một javabeans hay java object thông thường
   3. Backing beans phải được đăng ký với tập tin cấu hình (faces-config.xml) của ứng dụng để ứng dụng có thể cơ chế để lưu trữ và tương tác với bean
   4. Tương tác giữa form, UI component với Backing Beans sử dụng EL language với syntax #{EL expression} (dùng dấu # thay cho dấu $)
   5. Beans có khả năng hỗ trợ Biến cố - Event Handling, hỗ trợ chuyển trang – Navigation, thực hiện Validation
5. Conversion and validation
   1. Validator
      1. Hỗ trợ cơ chế kiểm tra dữ liệu trên các thành phần UI để đảm bảo dữ liệu được nhập đúng và có ràng buộc theo đúng yêu cầu của ứng dụng
   2. Converter
      1. Hỗ trợ cơ chế ép kiểu dữ liệu trong quá trình xử lý
   3. Message
      1. Hỗ trợ trình bày các thông tin – message về phía người dùng, thông báo lỗi hay hỗ trợ cơ chế I18N – hỗ trợ ứng dụng đa ngôn ngữ

**Presentation:**

Sau đây em xin được trình bày về **Thiết kế kiến trúc phần mềm (Software Design Architecture)**.

Nội dung gồm 4 phần chính: (Thiết kế kiến trúc) Architecture Presentation, Package Design, Database Design, Detail design flow.

**Architecture Presentation:**

Đây là mô hình thiết kế kiến trúc của hệ thống Carrier Trading Center. Được chia làm 3 layer chính, như trên sơ đồ. Presentation layer sử dụng nền tàng JSF (JavaServer Faces) theo mô hình MVC (Bean là model mở rộng thêm action, view là các page xhtml, faces servlet là controller giữ chức năng điều hướng, quản lý vòng đời của JSF). Business Object Layer sử dụng Spring tạo kết nối giữa các class thông qua cơ chế autowired. Data Access Layer, CTC sử dụng nền tảng Hibernate để làm việc với Hệ cơ sở dữ liệu My SQL. Và các thành phần Common dùng chung. Bên cạnh đó CTC tương tác với 2 thành phần bên ngoài là tích hợp với Cổng thanh toán ngân lương API để thực hiện nạp tiền cho User, và Sử dụng dịch vụ Gmail để thông báo thông tin trạng thái vận đơn cũng như tài khoản cho Users.

**Công nghệ chính CTC sử dụng là nền tảng JSF.**

Sàn thương mại CTC là hệ thống quản lý vận đơn đấu thầu, thông tin hàng hóa, người dùng … Xử lý trực tuyến các trạng thái đấu thầu … Cần phải validate tất cả dữ liệu input đầu vào, xử lý trạng thái của các UI control. Vì vậy qua tìm hiểu cũng như sự giúp đỡ của thầy hướng dẫn, bọn em chọn JSF là nền tảng phát triển hệ thống. JSF được đóng gói JS, html trong các thẻ, giúp lập trình viên đơn giản trong việc xây dựng giao diện người dùng, validate dữ liệu đầu vào, conversion thông tin hiển thị. Quản lý trạng thái UI control, đơn giản việc xử lý ajax qua các thẻ. JSF phát triển theo mô hình MVC, hỗ trợ xử lý sự kiện, từ view có thể gọi trực tiếp các action trong Bean. JSF hỗ trợ cơ chế rendering, điều khiển điều kiện hiển thị sau khi tính toán.

**Package Design:**

Bean – Model, xử lý thông tin Object, quản lý trạng thái.

BO – Layer xử lý businesslogic, kết nối DAO với Presentation.

DAO – Layer xử lý các tương tác với DB.

Utils, DTO là phần common, xử dụng chung cho các package. Constant, Mail, Data DTO…

Schedule là package các chức năng chạy ngầm của hệ thống, bảo gồm 3 công việc chính.

1. Thread tự động khớp lệnh.
2. Thread tự động send mail notification to users.
3. Thread tự động cập nhật giá tham chiếu.

**Database Design:**

Sau khi tìm hiểu, xây dựng đây là sơ đồ thiết kế DB của CTC – Bao gồm 14 bảng – User phân quyền theo roles (Admin, GoodsOwner, Carrier). BillOfLading – vận đơn, Carrier – Bảng đấu giá vận đơn.

**Detail Design flow:**

1. Screen Design: screen màn hình login.
2. Class diagram: gồm các class, interface được sử dụng trong chức năng login.
3. Sequence diagram: flow thực hiện chức năng login.



