**ÔN THI GIỮA KỲ LẬP TRÌNH WWW (JAVA)**

# LÝ THUYẾT

# SET UP

## IDE : IntelliJ

## Dependency :

* Gradle
* Mariadb
* Persistency

## Github Copilot

## Database : HeidiSQL – Mariadb

## Server : Tomcat / Payara

## Folder Structure (MVC) :



# THỰC HÀNH

## ĐỀ 1 :

**Yêu cầu**

Tạo một csdl có tên mydb với các bảng dữ liệu **account** (account\_id, full\_name, password, email, phone, status), bảng **role** (role\_id, role\_name, description, status). Một account thuộc về nhiều role, mỗi role có thể có nhiều account. Trong đó status là trường để chỉ trạng thái mẫu tin (1-active, 0-deactive, -1-xóa)

(Tip: tạo 1 bảng thứ ba có tên **grant\_access** (account\_id, role\_id, is\_grant, note) với trường is\_grant nhận giá trị 0-diasable, 1-enable.)

Một bảng có tên **log** để ghi vết lần đăng nhập. Thông tin bao gồm: id (auto-increment), account đăng nhập, ngày giờ đăng nhập, ngày giờ đăng xuất, ghi chú.

Tạo một jakartaEE project có tên **week01\_lab\_HotenSv\_mssv** (Có thể kết nối GitHub/GitLab để push code). Thực hiện các công việc sau:

* Tạo một servlet có tên **ControlServlet** (partern cùng tên). Servlet này nhận một tham số (parameter) có tên là **action**. Tham số này nhận các giá trị chuỗi để chỉ các hành động tương ứng.
* Kết nối với csdl, thực hiện các chức năng:
* Thêm, cập nhật, xóa ở các bảng đã cho.
* Đăng nhập
* Hiển thị thông tin tài khoản (nếu đăng nhập thành công)
* Hiển thị các quyền của một account.
* Hiển thị các account của một role
* Cấp quyền cho một account
* Ghi log mỗi lần account đăng nhập, đăng xuất.
* Một trang html hiển thị cửa sổ đăng nhập. Nếu đăng nhập thành công và là quyền admin thì hiển thị trang dashboard cho phép quản lý các account khác (bao gồm các quyền thêm, xóa, sửa và cấp quyền). Còn không (không phải admin) thì hiển thị thông tin của người đăng nhập cùng các quyền mà người đó có.





## ĐỀ 2 :

**Yêu cầu :**

**1. Xây dựng ứng dụng Web dùng Servlet/JSPs kết hợp với Database (tùy chọn SQL Server,**

**SQLite ... ) và Web server Tomcat thực hiện các yêu cấu theo mô hình MVC (Các Servlet đóng**

**vai trò Controller). Xây dựng Website để kinh doanh trao đổi thuốc trực tuyến. Website thực**

**hiện các chức năng cơ bản như xem danh sách các loại thuốc, xem danh sách thuốc, xem danh**

**sách thuốc theo từng loại, thêm thuốc mới.**

**2. Cơ sở dữ liệu được mô tà như sau:**

LOAITHUOC(MALOAI, TENLOAI)

THUOC(MATHUOC, TENTHUOC, GIA, NAMSX, MALOAI)

**Mỗi Table SV tự nhập 2 record dùng để kiểm tra chương trình.**

**3. Các thư viện jar sử dụng cho project, sinh viên tự download và cấu hình.**

**Layout Website minh hoa như sau:**

A close up of a paper

Description automatically generated

**YÊU CẦU:**

**Câu 1. (Iđ) Tạo cơ sở dữ liệu với các quan hệ như mô tả trên và nhập dữ liệu cần thiết để hiện thực chương trình.** Cơ sở dữ liệu trong DB lưu tên: MSSV\_HoTenSV

**Câu 2. (1đ) Tạo package chứa các class theo mô hình dữ liệu trên bao gồm Loaithuoc.java, Thuoc java (các phương thức get/set, contructors)**

**Câu 3. (2.5đ) Tạo package chứa các class QuanlyThuocDAO.java, QuanlyLoai ThuocDAO,java thực hiện kết nối đến database trên câu 1 thực hiện các phương thức CRUD theo các actions:**

a. (1.0đ) Xuất danh sách thuốc (getAllThuoc)

b. (0.5đ) Xuất danh sách các loại thuốc(getAllLoaiThuoc)

c. (0.5đ) Xuất danh sách thuc theo từng loại (getAllThuocByLoaiThuoc)

d. (0.5đ) Thêm thuốc mới. (addThuoc)

**Câu 4. (2.5đ) Tại package chứa class QuanlyThuocServlet.java hiện thực các actions trên câu 3**

a. (1.0đ) Xuất danh sách thuốc

b. (0.5đ) Xuất danh sách các loại thuốc

c. (0.5đ) Xuất danh sách thuốc theo từng loại thuốc

d. (0.5đ) Thêm thuốc mới.

**Câu 5. (3đ) Tạo View cho ứng dụng Web như sau:**

a. (1đ) Khi Website chạy hiển thị trang chứa toàn bộ danh sách thuốc, trên trang chứa action chuyển hướng sang trang **Xem danh sách các loại thuốc (câu b),** chứa action chuyển hướng sang trang **Thêm mới thuốc (câu d)**

b. (1đ) Xem danh sách các loại thuốc, hiển thị action chọn loại thuốc để hiện thị danh sách thuốc theo loại thuốc đang chọn (action tùy ý: link, button, listbox, ... ) (câu c)

c. (0.5đ) Xem danh sách thuốc theo từng loại thuốc.

d. (0.5đ) Thêm thuốc mới, tạo Form thêm thông tin thuốc với đầy đủ thông tin phù hợp với CSDL và Class Thuoc như mô tả, khi nhấn “Submit" thông tin thuốc được thêm vào CSDL và hiển thị danh sách thuốc gồm thuốc vừa thêm.

## ĐỀ 3 :

Xây dựng ứng dụng quản lý tin tức cho một website trực tuyến, các trang Web cần thực hiện các chức

năng: *xem tin tức theo từng danh mục, thêm mới tin tức và xóa tin tức không cần thiết.* Một phần của cơ

sở dữ liệu được mô tả như sau:

DANHMUC(MADM, TENDANHMUC, NGUOIQUANLY, GHICHU)

TINTUC(MATT, TIEUDE, NOIDUNGTT, LIENKET, MADM)

Xây dựng ứng dụng dùng Servlet/JSPs kết hợp với SQL Server và Web server Tomcat thực hiện các yêu

cầu với mô hình MVC (Các Servlet đóng vai trò Controller):

* Tạo cơ sở dữ liệu với các quan hệ như mô tả trên và nhập dữ liệu cần thiết để kiểm tra chương trình. Cơ sở dữ liệu trong SQL Server lưu tên: QUANLYDANHMUC
* Dùng HTML/CSS tạo Layout chung cho các *trang Web quản lý tin tức trực tuyến*



* Tạo lớp mô tả thông tin *TinTuc.java* (bao gồm các get/set cho các thuộc tính và các constructors) dùng để thao tác với tin tức trực tuyến.
* Tạo lớp *DanhSachTinTucQuanLy.java* (bao gồm các phương thức thao tác với CSDL: lấy tin tức theo danh mục tin tức, thêm mới tin tức và xóa tin tức).
* Tạo giao diện Web cho phép thực hiện thao tác liệt kê dữ liệu dùng JSPs: (*DanhSachTinTucServlet.java - DanhSachTinTuc.jsp*) Liệt kê dữ liệu danh sách tin tức theo từng danh mục.
* Tạo giao diện Web cho phép thực hiện giao diện thao tác với việc thêm dữ liệu
* Tạo Form cho phép thêm nội dung tin tức với đầy đủ thông tin phù hợp với CSDL (*TinTucFormServlet.java, TinTucForm.jsp*)
* Kiểm tra dữ liệu nhập phía Client trên Form:
* Mã TT, Tiêu đề, Liên kết, Nội dung, Thông tin danh mục của tin là bắt buộc nhập.
* Liên kết bắt đầu bởi “*http://”* (dùng RegularExpression).
* Nội dung không quá 255 ký tự (dùng RegularExpression).
* Thông tin nhập vào hợp lệ sau khi nhấn nút “Thêm” sẽ được thêm vào cơ sở dữ liệu và hiển thị dữ liệu trên màn hình (*KetQua.jsp hoặc trả về DanhSachTinTuc.jsp*).
* Tạo giao diện Web cho phép thực hiện chức năng quản lý: thao tác hủy dữ liệu. Khi chọn chức năng này, hiển thị danh sách và kèm theo chức năng hủy tin tức ra khỏi cơ sở dữ liệu

(*QuanLyFormServlet.java, QuanLyForm.jsp*).

## ĐỀ 4 :

Xây dựng ứng dụng quản lý thông tin sản phẩm cho một công ty mua bán điện thoại trực tuyến, các trang

Web cần thực hiện các chức năng: *xem danh sách sản phẩm theo nhà cung cấp, thêm mới sản phẩm và*

*xóa sản phẩm không cần thiết.* Một phần của cơ sở dữ liệu được mô tả như sau:

NHACUNGCAP(MANCC, TENNHACC, DIACHI, SODIENTHOAI)

DIENTHOAI(MADT, TENDT, NAMSANXUAT, CAUHINH, MANCC)

Xây dựng ứng dụng dùng Servlet/JSPs kết hợp với SQL Server và Web server Tomcat thực hiện các

công việc sau:

* Tạo cơ sở dữ liệu với các quan hệ như mô tả trên và nhập dữ liệu cần thiết để kiểm tra chương trình. Cơ sở dữ liệu trong SQL Server lưu tên: QUANLYDIENTHOAI
* Dùng HTML/CSS tạo Layout chung cho chương trình tương tự như sau:



* Tạo lớp mô tả thông tin *DienThoai.java, NhaCungCap.java* (bao gồm các get/set cho các thuộc tính và các constructors) dùng để thao tác với thông tin sản phẩm điện thoại.
* Tạo lớp *DanhSachDienThoaiQuanLy.java* (bao gồm các phương thức thao tác với CSDL: lấy sản phẩm theo mã (hoặc tên) nhà cung cấp, thêm mới và xóa sản phẩm điện thoại).
* Tạo giao diện Web cho phép thực hiện thao tác liệt kê dữ liệu dùng JSPs: (*DanhSachDienThoaiNCCServlet.java, DanhSachDienThoaiNCC.jsp*) Liệt kê dữ liệu danh sáchsản phẩm điện thoại theo từng nhà cung cấp.
* Tạo giao diện Web cho phép thực hiện giao diện thao tác với việc thêm dữ liệu
* Tạo Form thêm thông tin điện thoại với đầy đủ thông tin phù hợp với CSDL (*DienThoaiFormServlet.java, DienThoaiForm.jsp*)
* Kiểm tra dữ liệu nhập phía Client trên Form:
* Mã ĐT, Tên điện thoại, Năm sản xuất, Thông tin cấu hình: các thông tin này bắt buộcnhập.
* Năm sản xuất là số nguyên 4 chữ số (dùng RegularExpression).
* Thông tin cấu hình không quá 255 ký tự (dùng RegularExpression).
* Thông tin nhập vào hợp lệ sau khi nhấn nút “Thêm” sẽ được thêm vào cơ sở dữ liệu và hiển thị dữ liệu trên màn hình (*KetQua.jsp hoặc DanhSachDienThoaiNCC.jsp*).
* Tạo giao diện Web cho phép thực hiện chức năng quản lý: thao tác hủy dữ liệu. Khi chọn chức năng này, hiển thị danh sách và kèm theo chức năng hủy thông tin điện thoại ra khỏi cơ sở dữ liệu (*QuanLyFormServlet.java, QuanLyForm.jsp*).

## ĐỀ 5 :

**Yêu cầu :**

**1. Quản lý sản phẩm (3.0 điểm)**

* **(1.0 điểm)** Xây dựng REST API cho các chức năng CRUD (Create, Read, Update, Delete) với sản phẩm:
  + API để thêm sản phẩm mới (POST /products)
  + API để lấy danh sách sản phẩm (GET /products)
  + API để cập nhật thông tin sản phẩm (PUT /products/{id})
  + API để xóa sản phẩm (DELETE /products/{id})
* **(1.0 điểm)** Xây dựng giao diện JSP hiển thị danh sách sản phẩm, và form cho phép thêm, cập nhật sản phẩm:
  + Trang JSP hiển thị danh sách sản phẩm và nút "Sửa" hoặc "Xóa".
  + Form thêm sản phẩm mới với các trường như tên sản phẩm, giá, mô tả, nhà cung cấp.
* **(1.0 điểm)** Sử dụng **Servlet** để xử lý logic cho chức năng quản lý sản phẩm:
  + Servlet xử lý yêu cầu hiển thị danh sách sản phẩm.
  + Servlet xử lý yêu cầu thêm, sửa và xóa sản phẩm.

**Gợi ý về cấu trúc REST API:**

@Path("/products")

**public class ProductResource {**

**@POST**

**public Response createProduct(Product product) {**

**// Logic thêm sản phẩm**

**}**

**@GET**

**public List<Product> getAllProducts() {**

**// Logic lấy danh sách sản phẩm**

**}**

**@PUT @Path("/{id}")**

**public Response updateProduct(@PathParam("id") int id, Product product)**

**{**

**// Logic cập nhật sản phẩm**

**}**

**@DELETE @Path("/{id}")**

**public Response deleteProduct(@PathParam("id") int id)**

**{**

**// Logic xóa sản phẩm**

**}**

**}**

**2. Giỏ hàng và đặt hàng (3.0 điểm)**

* **(1.0 điểm)** Tạo chức năng giỏ hàng:
  + Người dùng có thể thêm sản phẩm vào giỏ hàng từ danh sách sản phẩm.
  + Hiển thị giỏ hàng với các sản phẩm đã chọn, kèm số lượng và giá.
* **(1.0 điểm)** Tạo REST API cho giỏ hàng và đơn đặt hàng:
  + API để thêm sản phẩm vào giỏ hàng (POST /cart)
  + API để hiển thị giỏ hàng (GET /cart)
  + API để tạo đơn đặt hàng từ giỏ hàng (POST /orders)
* **(1.0 điểm)** Xử lý đặt hàng:
  + Khi người dùng bấm nút "Đặt hàng", đơn đặt hàng sẽ được xử lý và lưu vào cơ sở dữ liệu.

**Gợi ý về cấu trúc REST API cho giỏ hàng và đặt hàng:**

@Path("/cart")

public class CartResource {

@POST

public Response addToCart(CartItem item)

{

// Logic thêm sản phẩm vào giỏ hàng

}

@GET

public Cart getCart()

{

// Logic lấy giỏ hàng hiện tại

}

}

@Path("/orders")

public class OrderResource {

@POST

public Response createOrder(Order order) {

// Logic xử lý đặt hàng

}

}

**3. WebSocket - Hỗ trợ trực tuyến (2.0 điểm)**

* **(1.0 điểm)** Sử dụng WebSocket để tạo chức năng chat trực tuyến giữa khách hàng và nhân viên hỗ trợ:
  + Khi khách hàng vào trang web, có thể sử dụng tính năng chat trực tiếp để nhận hỗ trợ từ nhân viên.
* **(1.0 điểm)** Xây dựng WebSocket server-side để nhận và xử lý các tin nhắn từ người dùng:
  + WebSocket server-side sẽ xử lý tin nhắn từ khách hàng và chuyển tiếp tới nhân viên hỗ trợ.
  + Tất cả tin nhắn sẽ được gửi và nhận qua giao diện chat thời gian thực.

**Gợi ý về cấu trúc WebSocket:**

@ServerEndpoint("/support")

public class SupportEndpoint { @OnOpen public void onOpen(Session session)

{ // Khi kết nối mở }

@OnMessage public void onMessage(String message, Session session)

{ // Khi nhận tin nhắn từ khách hàng session.getBasicRemote().sendText("Support: " + message); }

@OnClose public void onClose(Session session) { // Khi kết nối đóng } }

**4. Tìm kiếm và phân loại sản phẩm (2.0 điểm)**

* **(1.0 điểm)** Xây dựng chức năng tìm kiếm sản phẩm:
  + Cho phép người dùng tìm kiếm sản phẩm theo tên và loại sản phẩm.
  + Kết quả tìm kiếm sẽ được hiển thị trên trang JSP.
* **(1.0 điểm)** Tạo REST API cho chức năng tìm kiếm:
  + API để tìm kiếm sản phẩm theo tên (GET /products/search?name={name})
  + API để tìm kiếm theo loại sản phẩm (GET /products/search?category={category})

**Gợi ý về cấu trúc API tìm kiếm:**

@Path("/products/search")

public class SearchResource { @GET @Produces(MediaType.APPLICATION\_JSON)

public List<Product> searchProducts(@QueryParam("name") String name, @QueryParam("category") String category) { // Logic tìm kiếm sản phẩm theo tên hoặc loại } }