

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



**BÀI BÁO CÁO TIỂU LUẬN MÔN
LẬP TRÌNH WEB VÀ ỨNG DỤNG**

Đề tài: JAVA SERVLET & JSP

(Mô tả: Tìm hiểu Servlet & JSP và xây dựng trang web sử dụng Servlet & JSP)

Người hướng dẫn: **THẦY MAI VĂN MẠNH**

Người thực hiện: **TÔ VĨNH KHANG – 51800408**

NGUYỄN XUÂN THỊNH – 51800123

TRẦN CÔNG MINH - 51800701

Lớp : 18050203, 18050202, 18050402

Khoá : 22

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020

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



**BÀI BÁO CÁO TIỂU LUẬN MÔN
LẬP TRÌNH WEB VÀ ỨNG DỤNG**

Đề tài: JAVA SERVLET & JSP

(Mô tả: Tìm hiểu Servlet & JSP và xây dựng trang web sử dụng Servlet & JSP)

Người hướng dẫn: **THẦY MAI VĂN MẠNH**

Người thực hiện: **TÔ VĨNH KHANG – 51800408**

NGUYỄN XUÂN THỊNH – 51800123

TRẦN CÔNG MINH - 51800701

Lớp : 18050203, 18050202, 18050402

Khoá : 22

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn Khoa Công nghệ thông tin và Trường Đại học Tôn Đức Thắng đã tạo điều kiện cho chúng em được học tập trong suốt thời gian qua. Chân thành cảm ơn Thầy Mai Văn Mạnh đã giúp chúng em có thêm kiến thức về lập trình Web và các ứng dụng của môn học. Hiểu rõ hơn về cách làm việc của server. Do kiến thức về Java của chúng em còn hạn hẹp nên không tránh khỏi những thiếu sót trong quá trình thực hiện đề tài tiểu luận này. Thầy có thể góp ý để chúng em khắc phục và làm cho các trang web sau này sinh động hơn, chuyên nghiệp hơn cũng như mang lại nhiều giá trị có ích cho xã hội. Chân thành cảm ơn thầy.

TÓM TẮT

Tạo ra một trang web có sử dụng Java Servlet & JSP. Đây là webside demo cho việc thiết lập Servletlet JSP cho các công việc xử lý phía server , web cung cấp những tính năng như thêm xóa sửa dữ liệu, kiểm tra đăng nhập, thiết lập bài mật không cho phép người dùng vào trực tiếp trang nhưng trang phía sau nếu chưa login, đồng thời web cung cấp cơ chế session giúp ghi nhớ đăng nhập, khi người dùng đã đăng nhập rồi thì có thể vào trực tiếp trang mà không cần đăng nhập lại lần nữa

Tóm tắt sơ lược vấn đề nói trên:

Trang web hoạt động dựa theo nguyên tắc sau: khi có một yêu cầu đăng nhập ở trang login thì trang login sẽ chuyển yêu cầu đó đến trang filterAdmin để kiểm tra đăng nhập. sau khi đăng nhập thành công thì sẽ được chuyển hướng đến trang admin. Trang admin là trang chủ dùng để giới thiệu đôi nét về Servlet tiếp đến khi người dùng bấm vào đến trang quản lý thì sẽ chuyển đến trang quản lý trang quản lý sẽ hiện lên Database hiển thị danh sách các nhân viên trong một công ty, trang này có chức năng thêm xóa, sửa dữ liệu trong trường hợp cần thiết.

MỤC LỤC

LỜI CẢM ƠN.....	i
TÓM TẮT.....	ii
MỤC LỤC.....	1
DANH MỤC CÁC BẢNG, HÌNH.....	2
CHƯƠNG 1 – GIỚI THIỆU.....	4
Bảng phân công nhiệm vụ.....	4
CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT.....	7
2.1 Lý thuyết về Java Servlet.....	7
2.1.1 Java Servlet là gì?	7
2.1.2 Nhiệm vụ của Servlet là gì?	7
2.1.3 Nhược điểm của Servlet là gì?	7
2.2 Lý thuyết về JSP.....	8
2.2.1 JSP là gì?.....	8
2.2.2 JSP và Servlet.....	8
CHƯƠNG 3 – THỰC NGHIỆM.....	8
3.1 Chạy tay code.....	8
3.1.1 Phần giới thiệu chung.....	8
3.1.2 Phần đăng nhập và đăng kí tài khoản.....	9
3.1.3 Phần trang chủ và thêm xóa sửa dữ liệu trong danh sách.....	19
3.2 Sơ đồ usecase.....	29
CHƯƠNG 4 – KẾT QUẢ.....	30
4.1 Trang chủ.....	30
4.2 Trang quản lý dữ liệu trong database.....	31
TÀI LIỆU THAM KHẢO.....	34

DANH MỤC CÁC BẢNG, HÌNH

DANH MỤC HÌNH

Hình 3.1: Code để đăng nhập.....	9
Hình 3.2: Khung đăng nhập.....	10
Hình 3.3: Code tạo dữ liệu đăng nhập.....	10
Hình 3.4: Dữ liệu đăng nhập.....	11
Hình 3.5: Code cách tạo tài khoản.....	11
Hình 3.6: Tạo tài khoản.....	12
Hình 3.7: Code của Database.....	13
Hình 3.8: Code newAccount.....	15
Hình 3.9: Code thể hiện trang danh sách nhân viên công ty.....	20
Hình 3.10: Code trang insert.....	21
Hình 3.11: Code trang process_insert.....	21
Hình 3.12: Code của trang process_insert.....	23
Hình 3.13: Code của trang update.....	24
Hình 3.14: Code phần progress.....	24
Hình 3.15: Code phần delete.....	28
Hình 3.16: Sơ đồ usecase.....	29
 Hình 4.1: Trang chủ.....	 30
Hình 4.2: Trang quản lý danh sách nhân viên.....	31
Hình 4.3: Trang thêm dữ liệu nhân viên.....	31
Hình 4.4: Danh sách sau khi thêm mới nhân viên Tran Van Nam.....	32
Hình 4.5: Trang sửa dữ liệu nhân viên.....	32
Hình 4.6: Danh sách sau khi sửa dữ liệu của nhân viên Tran Van Nam.....	33
Hình 4.7: Danh sách sau khi xóa dữ liệu của nhân viên Tran Van Nam.....	33

DANH MỤC BẢNG

Bảng 1.1 Bảng phân công nhiệm vụ.....	5
Bảng 3.1: Ý nghĩa của class Database.....	14
Bảng 3. 2 Ý nghĩa của trang servlet.....	14
Bảng 3.3: Giải thích trang filterAdmin.....	16
Bảng 3.4: Giải thích trang CheckedLogin.....	17
Bảng 3.5: Giải thích tiếp trang FilterAdmin.....	18
Bảng 3.6: Đăng suất tài khoản.....	19
Bảng 3. 7: Giải thích code của phần thêm dữ liệu vào database.....	22
Bảng 3.8: Giải thích code của phần sửa dữ liệu trong database.....	26
Bảng 3. 9: Giải thích code của phần xóa dữ liệu trong database.....	27

CHƯƠNG 1 – GIỚI THIỆU

1. Thành viên nhóm:

Họ và tên: Tô Vĩnh Khang - 51800408 - Lớp: 18050203

Nguyễn Xuân Thịnh - 51800123 - Lớp: 18050202

Trần Công Minh - 51800701 - Lớp: 18050402

Bảng phân công nhiệm vụ

Họ và tên	Thời gian	Phân công nhiệm vụ
Tô Vĩnh Khang	14/2/2020-19/2/2020	Phân tích thiết kế yêu cầu , trình bày đặc tả use case , Phác họa bản thảo thiết kế cho giao diện web.
	19/2/2020-27/2/2020	Hiện thực giao diện trang web bằng HTML và CSS.
	1/3/2020-1/4/2020	Tiếp tục đưa ra ý tưởng mới cho tiểu luận và cải tiến giao diện. Trình bày 1 phần báo cáo tiểu luận.
	2/4/2020-9/4/2020	Tổng hợp kiểm tra và chỉnh sửa lại một số lỗi lần cuối.
	9/4/2020-10/4/2020	Kết thúc tiểu luận
Nguyễn Xuân Thịnh	14/2/2020-19/2/2020	Trao đổi về các đặc tả use case. Thảo luận vấn đề liên quan đến giao diện web
	19/2/2020-27/2/2020	Tìm hiểu về JSP, Servlet

	1/3/2020-1/4/2020	Tiếp tục đưa ra ý tưởng mới cho tiểu luận và cải tiến giao diện. Đồng thời sử dụng phần mềm và triển khai thực hiện code JSP.
	2/4/2020-6/4/2020	Trình bày 1 phần báo cáo tiểu luận.
	7/4/2020-9/4/2020	Kiểm tra và chỉnh sửa lại lần cuối.
	9/4/2020-10/4/2020	Kết thúc tiểu luận
Trần Công Minh	14/2/2020-19/2/2020	Thảo luận về giao diện web , đọc hiểu về cách thức trình bày báo cáo tiểu luận
	19/2/2020-27/2/2020	Tìm hiểu về Bảo mật web, JSP
	28/2/2020-1/3/2020	Trao đổi về việc kết hợp bảo mật vào trang web.
	2/3/2020-1/4/2020	Trình bày 1 phần báo cáo tiểu luận.
	2/4/2020-8/4/2020	Kiểm tra và chỉnh sửa lần cuối.
	9/4/2020-10/4/2020	Kết thúc tiểu luận.

Bảng 1.1 Bảng phân công nhiệm vụ.

2.Chi tiết quá trình thực hiện tiểu luận thực tế:

[14/02/2020] phân tích thiết kế yêu cầu bài tiểu luận , đọc hiểu cách thức trình bày file tiểu luận báo cáo.

[16/02/2020] đặc tả các use case hệ thống , cách thức hoạt động của web khi user sử dụng dụng.

[17/02/2020] thảo luận về giao diện web , tông màu , logo , ..

[18/02/2020] triển khai bằng bản phác họa thiết kế bằng tay

[19/02/2020] thực hiện triển khai bản phác họa lên web bằng HTML và CSS

[22/02/2020] tìm hiểu về JSP,Servlet và download phần mềm chạy code JSP.

[01/03/2020] bắt đầu thực hiện code JSP

[05/03/2020] nảy sinh vấn đề về việc phối hợp JSP với giao diện web đang được thiết kế.

[07/03/2020] khắc phục và phối hợp thành công giao diện web với JSP.

[08/03/2020] đan xen đó là viết một phần báo cáo tiểu luận về các lí thuyết đã tìm hiểu.

[09/03/2020] nảy ra một số ý tưởng mới cho việc thực hiện các giao thức sử dụng cho user.

[15/03/2020] trình bày demo một phần của trang web.

[17/03/2020] kết hợp được việc bảo mật vào trang web.

[20/03/2020] cải tiến một số chức năng để người dùng khi sử dụng cảm thấy thoải mái hơn.

[31/03/2020] báo cáo tiểu luận đã được trình bày được việc chạy code từng giai đoạn.

[02/04/2020] tổng hợp tất cả lại và kiểm tra về lỗi , cách trình bày file báo cáo , chính tả ,...

[09/04/2020] kiểm tra lần cuối cùng.

[10/04/2020] hoàn thành bài tiểu luận.

CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT

2.1 Lý thuyết về Java Servlet

2.1.1 Java Servlet là gì?

Là chương trình chạy trên một web hoặc ứng dụng máy chủ và hành động như một lớp trung gian giữa một yêu cầu đến từ một trình duyệt web hoặc HTTP client khác và cơ sở dữ liệu hoặc các ứng dụng trên máy chủ HTTP(HTTP Server)

Servlet có thể được mô tả bằng nhiều cách, tùy thuộc vào ngữ cảnh:

- Servlet là một công nghệ được sử dụng để tạo ra ứng dụng web
- Servlet là một API cung cấp interface và lớp bao gồm các tài liệu
- Servlet là một thành phần web được triển khai trên máy chủ để tạo ra web động.

Có nhiều interface và các lớp trong API servlet như Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse,...

2.1.2 Nhiệm vụ của Servlet là gì?

Servlet thực hiện các nhiệm vụ chính sau đây:

- Đọc dữ liệu cho client gửi. Gồm 1 mẫu HTML trên web hoặc cũng có thể từ một applet hoặc một chương trình khách hàng HTTP tùy chỉnh.
- Xử lý dữ liệu và tạo ra các kết quả.
- Gửi dữ liệu tới khách hàng.
- Gửi phản hồi HTTP ẩn cho khách hàng, điều này cần thiết lập cookie và tham số bộ nhớ đệm, và các tác vụ khác.

Ưu điểm của Servlet là: ta có thể thu thập đầu vào từ người dùng thông qua các hình thức trang web, hay từ cơ sở dữ liệu hoặc nguồn nào đó, và tạo ra được các trang web động

2.1.3 Nhược điểm của Servlet là gì?

Là phải viết cả code của Java và HTML trong một file tạo cảm giác rối rắm

2.2 Lý thuyết về JSP

2.2.1 JSP là gì?

JSP là viết tắt của tiếng Anh JavaServer Pages còn được biết đến với một cái tên khác là Java Scripting Preprocessor (có thể dịch là: Bộ tiền xử lý lệnh Java) là một công nghệ Java cho phép các nhà phát triển tạo nội dung HTML, XML hay một số định dạng khác của trang web một cách năng động. Công nghệ này cho phép người ta nhúng mã Java và một số hành động xử lý đã được định trước (pre-defined actions) vào trong nội dung tĩnh của trang

Trước khi hiển thị ra trình duyệt, tập tin JSP phải được biên dịch thành Servlet, dùng bộ biên dịch JSP (JSP compiler). Bộ biên dịch JSP có thể tạo servlet thành mã nguồn Java trước, rồi biên dịch mã nguồn ra tập tin .class dùng bộ biên dịch Java, hoặc có thể trực tiếp tạo mã byte code cho servlet từ trang JSP

2.2.2 JSP và Servlet

Ta xem JSP là một trừu tượng ở mức độ cao của Servlet, một phần mở rộng của Servlet

CHƯƠNG 3 – THỰC NGHIỆM

3.1 Chạy tay code

3.1.1 *Phần giới thiệu chung*

Ở tất cả các trang Servlet thay vì dùng Servlet mapping thì chúng em sử dụng Annotation nhằm giúp giảm thời gian code khi phải tạo thiết lập mapping cho từng Servlet và JSP .

Annotation được sử dụng để cung cấp thông tin dữ liệu cho mã Java. Nó là thông tin dữ liệu, Annotation không trực tiếp ảnh hưởng đến việc thực hiện các mã, mặc dù số loại chú thích thật sự có thể sử dụng cho mục đích đó.

Annotation được sử dụng với các mục đích như:

- Chỉ dẫn cho trình biên dịch (compiler)
- Chỉ dẫn trong thời điểm xây dựng (build-time)
- Chỉ dẫn trong thời gian chạy

3.1.2 Phần đăng nhập và đăng kí tài khoản

3.1.2.1 Phần đăng kí và đăng nhập

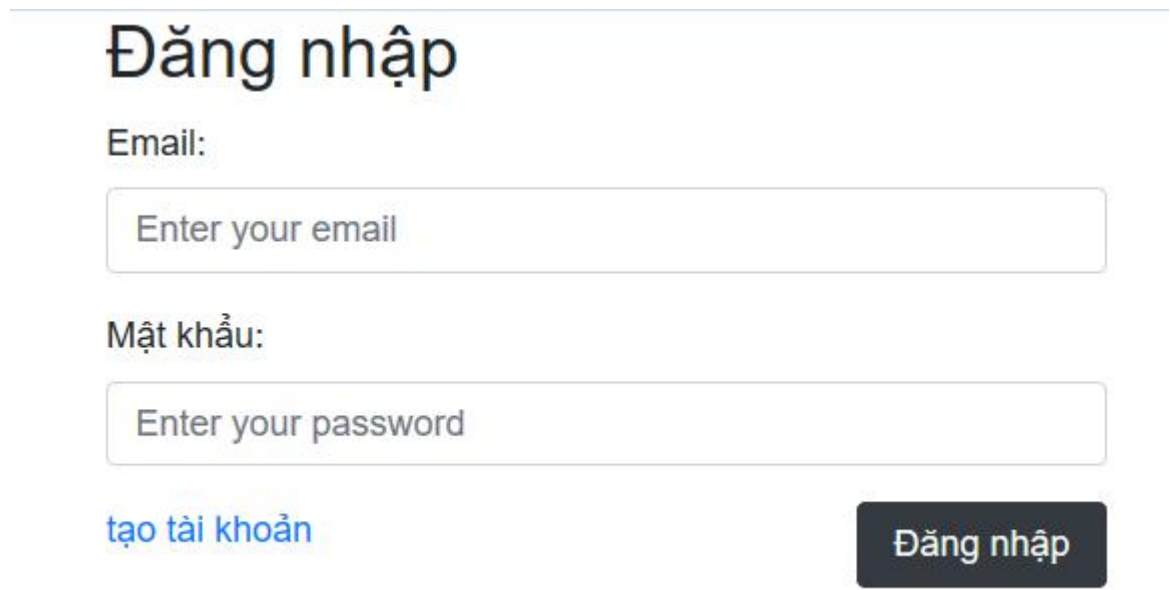
Đầu tiên khi người dùng muốn vào được trang chủ phải thông qua bước đăng nhập. Người dùng phải điền đúng tên tài khoản và mật khẩu đã lưu trong Database

```

<div class="container">
  <form action="/TieuLuan/web/adminPage.jsp" onsubmit="return CheckInput()" method="post">
    <h2>Đăng nhập</h2>
    <div class="form-group">
      <label for="email">Email:</label>
      <input type="text" onclick="ClearErrorMess()" class="form-control" id="email" placeholder="Enter your email" name="email">
    </div>
    <div class="form-group">
      <label for="pwd">Mật khẩu:</label>
      <input type="password" onclick="ClearErrorMess()" class="form-control" id="pwd" placeholder="Enter your password" name="pwd">
    </div>
    <p id="ErrorMess"></p>
    <div>
      <button type="submit" class="btn btn-dark" value="Login" name="action">Đăng nhập</button>
      <a href="/TieuLuan/web/newAccount.jsp">tạo tài khoản</a>
    </div>
  </form>
</div>

```

Hình 3.1: Code để đăng nhập



Đăng nhập

Email:

Mật khẩu:

[tạo tài khoản](#)

Hình 3.2: Khung đăng nhập

Form đăng nhập có phương thức là “POST” nhằm đảm bảo độ bảo mật, thông tin sẽ không được hiển thị trên thanh URL và sau khi bấm nút đăng nhập trang sẽ chuyển hướng qua trang adminPage.jsp

```
create database accounts;
use accounts;
create table login
(
    email varchar(40),
    password varchar(40),
    primary key (email)
);
```

Hình 3.3: Code tạo dữ liệu đăng nhập

email	password
khang@gmail.com	123456
minh@gmail.com	123456
minh123@	123456
thinh@gmail.com	123456
thinh12@	123456
thinh123@	123456
NULL	NULL

Hình 3.4: Dữ liệu đăng nhập

Nếu như dữ liệu người dùng không có trong database, người dùng có thể thêm vào bằng cách bấm “tạo tài khoản “ ở bên cạnh nút đăng nhập sau đó sẽ được chuyển hướng đến trang tạo tài khoản

```

<h2>tạo tài khoản</h2>
<div class=" form-group">
  <label> Email</label>
  <input onclick="deleteErrorE()" class="form-control" type = "text" name = "email" id ="email"placeholder="nhập email">
  <p id="errorEmail"></p>
</div>

<div class=" form-group">
  <label> Mật khẩu</label>
  <input onclick="deleteErrorP()" class="form-control" type = "password" name = "password" id ="password" placeholder="nhập mật khẩu ">
  <p id="errorPass"></p>
</div>

<div class=" form-group">
  <label> nhập lại mật khẩu</label>
  <input onclick="deleteErrorP2()" class="form-control" type = "password" name = "password2" id ="password2"placeholder="nhập lại mật khẩu ">
  <p id="errorPass2"></p>
</div>
<button type = "submit" class="btn btn btn-dark">LƯU </button>

```

Hình 3.5: Code cách tạo tài khoản

tạo tài khoản

Email

Mật khẩu

nhập lại mật khẩu

LƯU

Hình 3.6: Tạo tài khoản

Trong trang tạo tài khoản người dùng được yêu cầu nhập địa chỉ mail, mật khẩu và nhập lại mật khẩu ở bước này sẽ xác thực xem email, mật khẩu có hợp lệ hay không, mật khẩu nhập lần hai có đúng với mật khẩu nhập trước đó hay không.

- Email hợp lệ là email có chứa kí tự @ bên trong
- Mật khẩu hợp lệ là mật khẩu phải có ít nhất 6 kí tự
- Mật khẩu nhập lại hợp lệ khi có kí tự trùng với mật khẩu đăng nhập ở trên

(đưa hình từng điều kiện chạy trên trình duyệt vào + hình code js+ làm tên hình xuất tự động ko phải gõ nhé)

Người dùng bấm vào nút “LƯU” thì trang sẽ chuyển qua trang “**createNewAccount.java**” đây là trang servlet làm nhiệm vụ thêm dữ liệu vừa nhập ở trang “**newAccount.jsp**” vào database

Trước khi muốn tương tác với database cần phải mở kết nối đến với database mysql bằng một class tên là Database.java, được viết bên trong package com.database

```
package com.database;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Database {
    public static Connection getConnection() {

        String url = "jdbc:mysql://localhost:3306/accounts";
        String user= "root";
        String password="thinh54082166";
        try {
            Class.forName("com.mysql.jdbc.Driver");
            return DriverManager.getConnection(url,user,password);
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return null;
    }
}
```

Hình 3.7: Code của Database

Trong đó:

Code	Cách hoạt động
String url = "jdbc:mysql://localhost:3306/accounts"	Đây là địa chỉ kết nối với Mysql trong đó:jdbc là API, mysql là cơ sở dữ liệu, localhost là tên server, 3306 số cổng mysql đang dùng, accounts là tên của cơ sở dữ liệu
String user= "root"	Username để đăng nhập vào mysql được mặc định là root

String password="thinh54082166"	Password dùng để đăng nhập vào mysql
Class.forName("com.mysql.jdbc.Driver")	Đây là lớp Driver cho mysql
return DriverManager.getConnection(url,user,password);	Trả về phương thức dùng để kết nối mới mysql với 3 tham số truyền vào là (url, user, password)

Bảng 3.1: Ý nghĩa của class Database

Tại trang **createNewAccount** hàm sẽ được viết ở hàm doPost vì phương thức gửi lên là post. Tại đây sẽ thiết lập kết nối với database bằng cách gọi lại class Database.getConnection();

String sqlString= "insert into login values(?,?)";	Tạo một biến String dùng để chứa giá trị là một đoạn code dùng để thêm dữ liệu của mysql
PreparedStatement preparedStatement; preparedStatement connection.prepareStatement(sqlString);	Khởi tạo PreparedStatement. PreparedStatement được sử dụng để thực thi các câu lệnh truy vấn
String email= request.getParameter("email"); String password= request.getParameter("password");	Dùng để gửi yêu cầu lên server nhằm lấy giá trị “name” là “email” và “password” đã được gửi đi từ trang newAccounts.jsp
preparedStatement.setString(1, email); preparedStatement.setString(2,password);	Gắn dữ liệu vào preparedStatement với 1 và 2 là vị trí cần gắn giá trị, email , password là giá trị gắn vào vị trí 1 , 2 tương ứng
int i=preparedStatement.executeUpdate()	Câu lệnh thực hiện truy vấn (dùng để create drop insert update, delete), kiểu dữ liệu của executeUpdate() là int.
response.sendRedirect("/TieuLuan/web/Login.jsp")	Chuyển trang đến trang login.jsp sau khi hoàn tất việc lưu dữ liệu mới vào database

Bảng 3. 2 Ý nghĩa của trang servlet

```

<form action="/TieuLuan/CreateNewAccount" method="post" onsubmit="return checkInput()">
  <h2>tạo tài khoản</h2>
  <div class=" form-group">
    <label> Email</label>
    <input onclick="deleteErrorE()" class="form-control" type = "text" name = "email" id ="email"placeholder="nhập email">
    <p id="errorEmail"></p>
  </div>

  <div class=" form-group">
    <label> Mật khẩu</label>
    <input onclick="deleteErrorP()" class="form-control" type = "password" name = "password" id ="password" placeholder="nhập mật khẩu ">
    <p id="errorPass"></p>
  </div>
  <div class=" form-group">
    <label> nhập lại mật khẩu</label>
    <input onclick="deleteErrorP2()" class="form-control" type = "password" name = "password2" id ="password2"placeholder="nhập lại mật khẩu ">
    <p id="errorPass2"></p>
  </div>
  <button type = "submit" class="btn btn btn-dark">LƯU </button>
</form>

```

Hình 3.8: Code newAccount

Sau khi trang được chuyển lại trang “Login.jsp” bây giờ người dùng có thể đăng nhập vào bằng tài khoản vừa tạo. Hoàn tất phần nhập “email” và “password” bấm đăng nhập thì trang sẽ chuyển đến trang “adminPage.jsp”. Nhưng trước khi được phép vào trang “adminPage.jsp”.thì phải đi qua filter có tên là “filterAdmin.java” đây là trang dùng để bảo mật,dùng để chặn các yêu cầu từ một Client trước khi chúng truy cập một nguồn tại backend ⁽⁴⁾ không cho phép người dùng có thể đi trực tiếp vào các trang bên trong như “adminPage.jsp” , “EmployeeManagement.jsp”, “Insert.jsp”, “update.jsp”

HttpServletRequest req = (HttpServletRequest) request;	Ép kiểu request và response từ servletRequest và ServletResponse sang
HttpServletResponse res =	HttpServletRequest và HttpServletResponse để sử dụng được các phương thức định sẵn của

(HttpServletResponse) response;	HttpServlet
HttpSession httpSession = req.getSession()	Khởi tạo session
Object object = httpSession.getAttribute("email");	Trả về đối tượng đã được kết nối với name là "email" hoặc trả về null nếu không có đối tượng nào được kết nối với "email".
if (object != null) { chain.doFilter(request, response);}	Nếu email có giá trị thì cho phép đi tới trang đích, nếu không có giá trị thì sẽ xét điều kiện trong else
String email = request.getParameter("email"); String password = request.getParameter("pwd");	Gửi yêu cầu lấy giá trị của name có tên là "email" và password" được gửi lên server
CheckedLogin temp = new CheckedLogin();	CheckedLogin là một class được viết trong package com.database đây là class dùng để kiểm tra dữ liệu của người dùng nhập vào có giống trong database có hay không.
boolean isLogin = temp.check(email, password);	Khởi tạo 1 biến có giá trị là boolean gọi lại hàm check với tham số truyền vào là email và password

Bảng 3.3: Giải thích trang filterAdmin

Chuyển qua class CheckedLogin

Trong class này có 1 hàm là check với kiểu dữ liệu là boolean, tham số truyền vào là kiểu String bao gồm email và password

<pre>Connection connection = Database.getConnection();</pre>	<p>Kết nối với cơ sở dữ liệu</p> <p>Tạo biến chứa code của cơ sở dữ liệu</p>
<pre>String sql = "select * from login where email=? and password=?"; PreparedStatement ps; ps = (PreparedStatement) connection.prepareStatement(sql) ps.setString(1, email; ps.setString(2, password);</pre>	<p>Khởi tạo PreparedStatement.</p> <p>PreparedStatement được sử dụng để thực thi các câu lệnh truy vấn và gán giá trị email, password vào vị trí tương ứng</p>
<pre>ResultSet rs = ps.executeQuery();</pre>	<p>Thực hiện truy vấn chọn trả về "rs"</p>
<pre>if(rs.next()) { connection.close(); return true;} return false;</pre>	<p>Kiểm tra lần lượt từng dữ liệu xem nếu có dữ liệu trùng khớp với dữ liệu trong database thì đóng kết nối trả về true nếu không trả về false</p>

Bảng 3.4: Giải thích trang CheckedLogin

Trở lại filterAdmin.java

<pre>if (isLogin == true && email!=null && password!=null) { HttpSession.setAttribute("email", email); chain.doFilter(request, response);} </pre>	<p>Nếu dữ liệu trả về giá trị là true và email, password lấy ra từ request.getParameter có dữ liệu thì sẽ lưu giá trị vào session và cho phép người dùng đi đến những trang phía sau</p>
<pre>Else res.sendRedirect("/TieuLuan/web/Login.jsp"); </pre>	<p>Nếu không thì sẽ quay trở lại trang Login.jsp</p>

Bảng 3.5: Giải thích tiếp trang FilterAdmin

Giải thích cách hoạt động của trang “CheckedLogin.java”:

Khi bên filterAmind gọi đến hàm check thì nó sẽ lấy giá trị “email” và “password” đã được lấy ở phương thức getParameter để truyền vào dấu chấm hỏi(?) trong String sql trong class check. Sau khi thực gán giá trị thực thi cho “rs” ,” rs” sẽ chạy trong database lần lượt và so sánh khi nào gặp dữ liệu giống với “email” và “password” truyền vào thì sẽ dừng lại đóng kết nối trả về true. ở bên filterAdmin nhận được giá trị true trong “isLogin” và cùng với điều kiện là “email”, “password” có giá trị thì sẽ gán giá trị email vào session để cho lần truy cập sau không cần phải đăng nhập lại, cuối cùng chuyển đến trang đích

3.1.2.2 Phần đăng xuất tài khoản

Trong trang adminPage ở phần menu ngang sẽ có chữ “sign out” để thoát đăng nhập (gỡ bỏ session đã lưu). Khi Người dùng bấm vào “sign out” thì server sẽ chuyển thông điệp đến trang “Logout.java” trong packed “com.tieuluan”. Đầu tiên thiết lập annotation

Trong trang adminPage ở phần menu ngang sẽ có chữ “sign out” để thoát đăng nhập

@WebServlet(urlPatterns = {"/Logout"})	Để thay thế cho servlet mapping , tạo đường dẫn trên url.
HttpSession httpSession=request.getSession();	Đầu tiên khởi tạo lại session
httpSession.removeAttribute("email");	Xóa giá trị đã được lưu trong session bằng lệnh removeAttribute()
response.sendRedirect("/TieuLuan/web/Login.jsp");	Chuyển hướng trang về lại trang “Login.jsp”

Bảng 3.6: Đăng xuất tài khoản

3.1.3 Phần trang chủ và thêm xóa sửa dữ liệu trong danh sách

3.1.3.1 Giới thiệu chung

Tại trang adminPage trên trình duyệt, bấm vào “ đến trang quản lý” thì sẽ được chuyển đến trang “EmployeeManagement” đây là trang hiện danh sách của các nhân viên trong công ty


```

<form action=/TieuLuan/process_insert id="form" method="post">
  <div class="form-group">
    MaNV: <input type="text" name="MaNV" class="form-control">
  </div>
  <div class="form-group">
    tenNV: <input type="text" name="tenNV" class="form-control">
  </div>
  <div class="form-group">
    Chucvu: <input type="text" name="ChucVu" class="form-control">
  </div>
  <div class="form-group">
    taikhoan: <input type="text" name="taikhoan" class="form-control">
  </div>
  <div class="form-group">
    matkhau: <input type="password" name="matkhau" class="form-control">
  </div>
  <button class="btn btn-success" type="submit">submit</button>
</form>

```

Hình 3.10: Code trang insert

Sẽ có năm mục để người dùng nhập vào là MaNV, “TenNV”, “Chucvu”, “taikhoan”, “matkhau” sau khi điền đủ 5 mục người dùng bấm vào “submit” thì trang sẽ chuyển dữ liệu vừa nhập qua trang “process_insert.java” để thêm dữ liệu vào trong database

```

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    Connection connection = Database.getConnection();
    String sqlString= "insert into nhanvien values(?, ?, ?, ?, ?)";
    String MaNV= request.getParameter("MaNV");
    String tenNV= request.getParameter("tenNV");
    String ChucVu= request.getParameter("ChucVu");
    String taikhoan= request.getParameter("taikhoan");
    String matkhau= request.getParameter("matkhau");
    int i;
    if(MaNV!=null)
    try {
        PreparedStatement pt= connection.prepareStatement(sqlString);
        pt.setString(1, MaNV);
        pt.setString(2, tenNV);
        pt.setString(3, ChucVu);
        pt.setString(4, taikhoan);
        pt.setString(5, matkhau);
        i=pt.executeUpdate();
        response.sendRedirect("/TieuLuan/web/EmployeeManagement.jsp");
    }
}

```

Hình 3.11: Code trang process_insert

Tại phần này cách hoạt động giống với thêm tài khoản mật khẩu vào database

Connection connection = Database.getConnection();	Thiết lập kết nối với database
String sqlString= "insert into nhanvien values(?,?,?,?);"	Tạo biến chứa code cơ sở dữ liệu thêm
String MaNV= request.getParameter("MaNV"); String tenNV= request.getParameter("tenNV"); String ChucVu= request.getParameter("ChucVu"); String taikhoan= request.getParameter("taikhoan"); String matkhau= request.getParameter("matkhau");	Gửi yêu cầu lấy giá trị của “MaNV”, “TenNV”, “ChucVu”, “taikhoan”, “matkhau” lên server và gán vào các biến tương ứng
PreparedStatement pt= connection.prepareStatement(sqlString);	Khởi tạo PreparedStatement
pt.setString(1, MaNV); pt.setString(2, tenNV); pt.setString(3, ChucVu); pt.setString(4, taikhoan); pt.setString(5, matkhau);	Gán giá trị đã lấy được từ server vào từng vị trí tương ứng trong database
i=pt.executeUpdate();	Câu lệnh thực hiện truy vấn , trong trường hợp này là thêm dữ liệu cho bảng.
response.sendRedirect("/TieuLuan/web/Employ eeManagement.jsp");	Điều hướng về lại trang EmployeeManagement để hoàn tất việc thêm danh sách

Bảng 3. 7: Giải thích code của phần thêm dữ liệu vào database

Mặc dù thông tin của method trong form của Insert.jsp là post nhưng cũng có thể ghi ở hàm doGet bên process_insert.java sau đó gọi lại làm hàm ở doPost

```
/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    Connection connection = Database.getConnection();
    String sqlString= "insert into nhanvien values(?, ?, ?, ?, ?)";
    String MaNV= request.getParameter("MaNV");
    String tenNV= request.getParameter("tenNV");
    String ChucVu= request.getParameter("ChucVu");
    String taikhoan= request.getParameter("taikhoan");
    String matkhou= request.getParameter("matkhou");
    int i;
    if(MaNV!=null)
    try {
        PreparedStatement pt= connection.prepareStatement(sqlString);
        pt.setString(1, MaNV);
        pt.setString(2, tenNV);
        pt.setString(3, ChucVu);
        pt.setString(4, taikhoan);
        pt.setString(5, matkhou);
        i=pt.executeUpdate();
        response.sendRedirect("/TieuLuan/web/EmployeeManagement.jsp");
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
    doGet(req, resp);
}
```

Hình 3.12: Code của trang process_insert

3.1.3.3 Cập nhật (UPDATE)

Người dùng bấm vào nút UPDATE tại vị trí muốn cập nhật, chỉnh sửa thì trang sẽ gửi tất cả thông tin của của cá nhân mà người dùng muốn chỉnh sửa qua trang “process.java”

```

<body>
<%
    Connection connection = Database.getConnection();
    try {
        Statement statement = connection.createStatement();
        String MaNV = request.getParameter("MaNV");
        String sql = "select * from nhanvien where MaNV='" + MaNV + "'";
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            <%
                <form action="/TieuLuan/process" method="post" id="form">
                    <div class="form-group">
                        MaNV: <input type="text" name="MaNV" value="<%=resultSet.getString("MaNV")%>" class="form-control">
                    </div>
                    <div class="form-group">
                        tenNV: <input type="text" name="tenNV" value="<%=resultSet.getString("tenNV")%>" class="form-control">
                    </div>
                    <div class="form-group">
                        Chucvu: <input type="text" name="ChucVu" value="<%=resultSet.getString("ChucVu")%>" class="form-control">
                    </div>
                    <div class="form-group">
                        taikhoan: <input type="text" name="taikhoan" value="<%=resultSet.getString("taikhoan")%>" class="form-control">
                    </div>
                    <div class="form-group">
                        matkhu: <input type="text" name="matkhu" value="<%=resultSet.getString("matkhu")%>" class="form-control">
                    </div>
                    <button class="btn btn-success" type="submit">submit</button>
                </form>
            <%
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        <%
    }
    <%
%>

```

Hình 3.13: Code của trang update

```

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    Connection connection = Database.getConnection();
    String MaNV = request.getParameter("MaNV");
    String tenNV = request.getParameter("tenNV");
    String ChucVu = request.getParameter("ChucVu");
    String taikhoan = request.getParameter("taikhoan");
    String matkhu = request.getParameter("matkhu");
    String sql = "update nhanvien set MaNV=?, tenNV=?, ChucVu=?, taikhoan=?, matkhu=? where MaNV='" + MaNV + "'";
    //String sql = "update nhanvien set MaNV='" + MaNV + "', tenNV='" + tenNV + "', ChucVu='" + ChucVu + "', taikhoan='" + taikhoan + "'";
    try {
        PreparedStatement pt = connection.prepareStatement(sql);
        pt.setString(1, MaNV);
        pt.setString(2, tenNV);
        pt.setString(3, ChucVu);
        pt.setString(4, taikhoan);
        pt.setString(5, matkhu);
        int i = pt.executeUpdate();
        if (i > 0) {
            System.out.print("data updated successfully!");
        }
        else {
            System.out.print("data update false");
        }
        response.sendRedirect("/TieuLuan/web/EmployeeManagement.jsp");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

Hình 3.14: Code phần progress

<pre>Connection connection = Database.getConnection();</pre>	Thiết lập kết nối với database
<pre>String MaNV = request.getParameter("MaNV"); String tenNV = request.getParameter("tenNV"); String ChucVu= request.getParameter("ChucVu"); String taikhoan= request.getParameter("taikhoan"); String matkhau= request.getParameter("matkhau");</pre>	Gửi yêu cầu lấy giá trị có name lần lượt là “MaNV”, “TenNV”, “ChucVu”, “taikhoan”, “matkhau” lên server và gắn vào các biến tương ứng
<pre>String sql="update nhanvien set MaNV=?, tenNV=?,ChucVu=?,taikhoan=?,matkhau =? where MaNV='"+MaNV+"'";</pre>	Tạo câu lệnh sql thực hiện việc update dữ liệu tại nơi có MaNV cần được cập nhật
<pre>PreparedStatement pt = connection.prepareStatement(sql); pt.setString(1, MaNV);</pre>	Khởi tạo PreparedStatement
<pre>pt.setString(2, tenNV); pt.setString(3, ChucVu); pt.setString(4, taikhoan); pt.setString(5, matkhau);</pre>	Gắn giá trị đã lấy được từ server vào từng vị trí tương ứng trong database
<pre>int i =pt.executeUpdate(); if(i>0) {</pre>	câu lệnh thực hiện truy vấn, trong trường hợp này là chỉnh sửa dữ liệu cho

<pre> System.out.print("data updated successfully!"); } else { System.out.print("data update false"); } </pre>	bảng. Với $i > 0$ thì cập nhật thành công còn lại thì thất bại
<pre> response.sendRedirect("/TieuLuan/web/E mployeeManagement.jsp"); </pre>	Điều hướng về lại trang EmployeeManagement để hoàn tất việc chỉnh sửa danh sách
<pre> Connection connection = Database.getConnection(); </pre>	Thiết lập kết nối với database
<pre> Statement statement = connection.createStatement(); </pre>	Khởi tạo statement
<pre> String MaNV = request.getParameter("MaNV"); String sql = "select * from nhanvien where MaNV =" + MaNV + """; </pre>	Gửi yêu cầu lên server để lấy giá trị có name là “MaNV” sau đó gán câu lệnh chọn nhân viên của sql vào biến “sql”
<pre> ResultSet resultSet = statement.executeQuery(sql); while (resultSet.next()) { </pre>	Thực thi câu lệnh vì truy suất dữ liệu lần lượt

Bảng 3.8: Giải thích code của phần sửa dữ liệu trong database

3.1.3.4 Xóa (*DELETE*)

Ở giao diện EmployeeManagement trên trình duyệt khi người dùng bấm nút “DELETE” thì sẽ được chuyển sang trang “delete.java” trong packed “com.tieuluan” để thực hiện việc xóa dữ liệu

Connection connection = Database.getConnection();	Thiết lập kết nối với database
String MaNV= request.getParameter("MaNV"); String sql="delete from nhanvien where MaNV='"+MaNV+"'";	Gửi yêu cầu lấy giá trị có name là “MaNV” lên server và gắn vào biến MaNV Tạo biến chứa code xóa trong trong mysql
PreparedStatement preparedStatement= connection.prepareStatement(sql);	Khởi tạo PreparedStatement
i= preparedStatement.executeUpdate();	Thực hiện câu lệnh truy vấn xóa dữ liệu
response.sendRedirect("/TieuLuan/web/Employee Management.jsp");	Điều hướng về lại trang EmployeeManagement để hoàn tất việc xóa nhân viên ra khỏi danh sách

Bảng 3. 9: Giải thích code của phần xóa dữ liệu trong database

```

/**
 * Servlet implementation class delete
 */
@WebServlet(urlPatterns = {"/delete"})
public class delete extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public delete() {
        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        Connection connection = Database.getConnection();

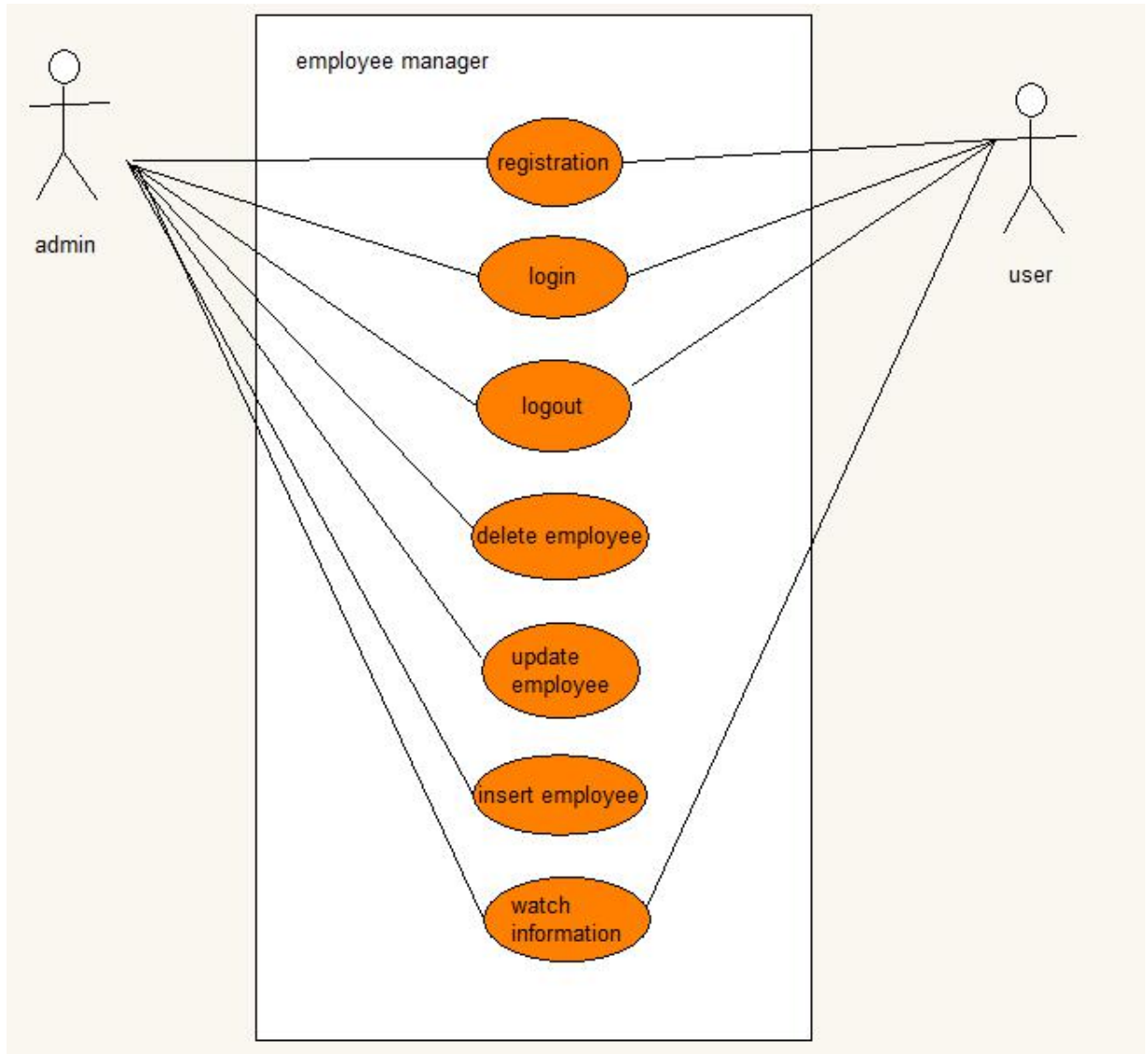
        String MaNV= request.getParameter("MaNV");
        PrintWriter printWriter = response.getWriter();
        printWriter.print(MaNV);
        int i;
        try {
            Statement statement = connection.createStatement();
            i= statement.executeUpdate("delete from nhanvien where MaNV='"+MaNV+"'");// VÌ là kiểu String nên phải nằm trong n
            response.sendRedirect("/TieuLuan/web/EmployeeManagement.jsp");
            System.out.println("Data Deleted Successfully!");
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        doGet(request, response);
    }
}

```

Hình 3.15: Code phần delete

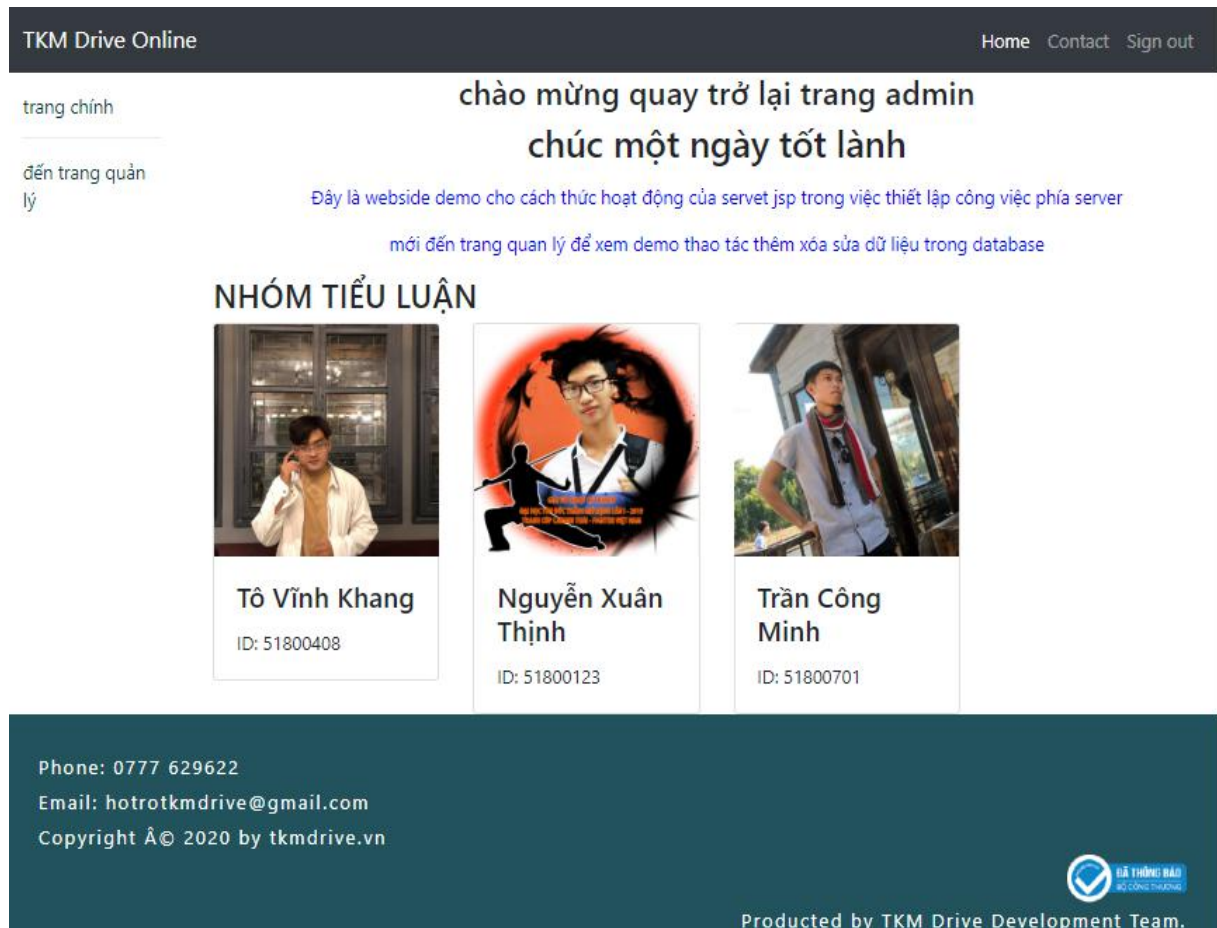
3.2 Sơ đồ usecase



Hình 3.16: Sơ đồ usecase

CHƯƠNG 4 – KẾT QUẢ

4.1 Trang chủ



Hình 4.1: Trang chủ

4.2 Trang quản lý dữ liệu trong database

DANH SÁCH NHÂN VIÊN TRONG CÔNG TY						
id	tên nhân viên	chức vụ	tài khoản	mật khẩu	thay đổi nhân viên	xóa nhân viên
TK	Nguyễn Bích Vân	Thư kí	nguyenbichvan	12345	<button>UPDATE</button>	<button>DELETE</button>
TP	Nguyễn Gia Lâm	trưởng phòng	nguyengialam	12345	<button>UPDATE</button>	<button>DELETE</button>

thêm nhân viên

ADD

BACK

Hình 4.2: Trang quản lý danh sách nhân viên

THÊM NHÂN VIÊN

MaNV:

tenNV:

Chucvu:

taikhoan:

matkhau:

submit

Hình 4.3: Trang thêm dữ liệu nhân viên

DANH SÁCH NHÂN VIÊN TRONG CÔNG TY						
id	tên nhân viên	chức vụ	tài khoản	mật khẩu	thay đổi nhân viên	xóa nhân viên
LC	Tran Van Nam	trưởng phòng	nam@gmail.com	123456	<button>UPDATE</button>	<button>DELETE</button>
TK	Nguyễn Bích Vân	Thư kí	nguyenbichvan	12345	<button>UPDATE</button>	<button>DELETE</button>
TP	Nguyễn Gia Lâm	trưởng phòng	nguyengialam	12345	<button>UPDATE</button>	<button>DELETE</button>

thêm nhân viên

ADD

BACK

Hình 4.4: Danh sách sau khi thêm mới nhân viên Tran Van Nam

MaNV:

LC

tenNV:

Tran Van Nam

Chucvu:

lao cong

taikhoan:

nam@gmail.com

matkhau:

.....

submit

Hình 4.5: Trang sửa dữ liệu nhân viên

DANH SÁCH NHÂN VIÊN TRONG CÔNG TY						
id	tên nhân viên	chức vụ	tài khoản	mật khẩu	thay đổi nhân viên	xóa nhân viên
LC	Tran Van Nam	lao cong	nam@gmail.com	123456	UPDATE	DELETE
TK	Nguyễn Bích Vân	Thư kí	nguyenbichvan	12345	UPDATE	DELETE
TP	Nguyễn Gia Lâm	trưởng phong	nguyengialam	12345	UPDATE	DELETE

thêm nhân viên

ADD

BACK

Hình 4.6: Danh sách sau khi sửa dữ liệu của nhân viên Tran Van Nam

DANH SÁCH NHÂN VIÊN TRONG CÔNG TY						
id	tên nhân viên	chức vụ	tài khoản	mật khẩu	thay đổi nhân viên	xóa nhân viên
TK	Nguyễn Bích Vân	Thư kí	nguyenbichvan	12345	UPDATE	DELETE
TP	Nguyễn Gia Lâm	trưởng phong	nguyengialam	12345	UPDATE	DELETE

thêm nhân viên

ADD

BACK

Hình 4.7: Danh sách sau khi xóa dữ liệu của nhân viên Tran Van Nam

TÀI LIỆU THAM KHẢO

[1] <https://viettuts.vn/java-jdbc/preparedstatement-trong-java>

Thời gian truy cập 22g10 ngày 2/4/2020

[2] <https://stackjava.com/network-programming/sanh-statement-voi-preparedstatement-callablestatement-trong-jdbc-java.html>

Thời gian truy cập 22g15 ngày 2/4/2020

[3] <https://o7planning.org/vi/10197/huong-dan-java-annotation/>

Thời gian truy cập 23g13 ngày 2/4/2020

[4] https://www.vietjack.com/jsp/filter_trong_jsp.jsp

Thời gian truy cập 20h02 ngày 03/04/2020

[5] <https://viettuts.vn/servlet>

Thời gian truy cập 23h10 ngày 03/04/2020

[6] <https://vi.wikipedia.org/wiki/JSP>

Thời gian truy cập 8h30 ngày 04/04/2020