Bai 6: Tạo chức năng đăng nhập

-ExcuteNonquery chỉ trả về số dòng của các chức năng insert, delete, update

-Nếu muốn kiểu tra mật khẩu đăng nhập có đúng không thì phải kiểm tra bằng cách trả về 1 Datatable rồi đếm xem trong Datatable có bao nhiêu dòng, nêu > 0 thì đăng nhập thành công9vì tài khoản và mật khẩu đã nhập trùng với 1 dòng trong cơ sở dữ liệu!)

Bài 7: hạn chế lỗi SQL injection

-lỗi đăng nhập mà không cần tài khoản mật khẩu:

C1: Sử dụng stored procedure trong sql

Cach tao procedure:

create proc usp\_login --user’s procedure

@username nvarchar(100),

@password nvarchar(100)

as

begin

Select \* from dbo.account where @username = username and @password = password

End

Bai 8: Hien thi danh sach ban an

TableDAO: Xử lý lấy dữ liệu từ CSDL lên, tạo 1 đối tượng static instance để lớp view gọi đến

Có 1 hàm loadTableList để load danh sach các table từ csdl lên

Table(trong thư mục DTO): lưu trữ các thuộc tính của đối tượng table

Các bước của phần này:

+Tạo lớp Table với các thuộc tính bình thường và có thêm hàm khởi tạo, lưu ý óc thêm

public Table(DataRow row)

{

this.ID = (int)row["id"];

this.Name = row["name"].ToString();

this.Status = row["Status"].ToString();

}

+tạọ TableDAO và singleton cho nó, sau đó viết thêm hàm loadTableList, mỗi row trong cơ sở dữ liệu sẽ tương ứng với 1 đối tượng table được tạo mới cho lớp Table

public List<Table> loadTableList()

{

List<Table> tablelst = new List<Table>();

DataTable data = DataProvider.Instance.ExcecuteQuery("exec usp\_GetTableList");

foreach(DataRow row in data.Rows)

{

Table table = new Table(row);

tablelst.Add(table);

}

return tablelst;

}

+Ở tầng view, Viết hàm loadTableList để gọi thằng loadTableList của thằng TableDAO lên và tạo ra các button tương ứng với mỗi bàn trong DataTable, set kích thước cho button sao cho hợp ní