CÔNG NGHỆ WEB

THIẾT KẾ MỘT SỐ CHỨC NĂNG WEB

NỘI DUNG

- Một số nguyên tắc thiết kế
- Kỹ thuật giỏ hàng
- Đăng nhập, phân quyền, bảo mật
- Quản trị nội dung

MỘT SỐ NGUYÊN TẮC THIẾT KẾ

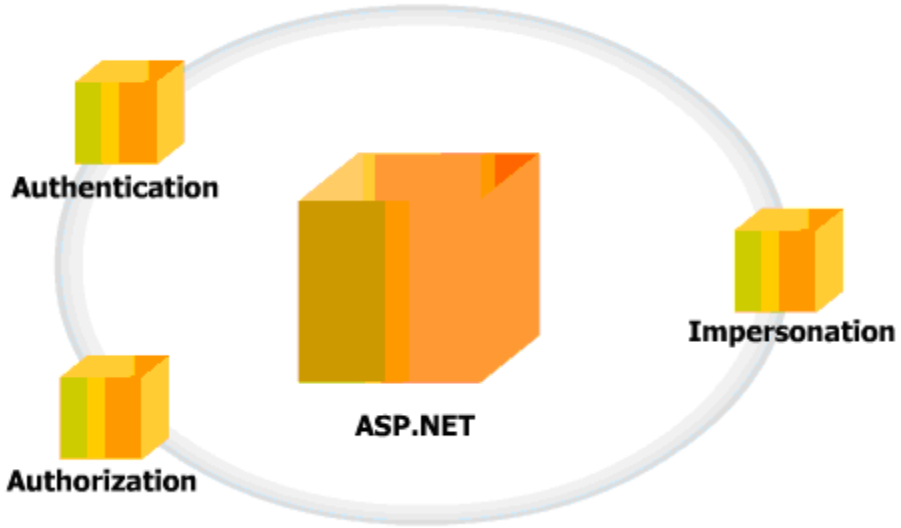
- Tổ chức website chặt chẽ và dễ sử dụng.
- Dễ dàng khám phá các đường link
- Dễ theo dõi quá trình tương tác
- Nội dung không nhiều hình ảnh
- Sử dụng từ ngữ dễ hiểu
- Tương thích với đa số trình duyệt web.

Kỹ thuật giỏ hàng

- Kỹ thuật giỏ hàng
  - Giỏ hàng là gì
  - Ứng dụng
  - Xây dựng

SECURITY BASICS

- Đó là các khái niệm:
  - identity (Nhận dạng),
  - authentication (Xác thực),
  - và authorization (Quyền hạn):



SECURITY BASICS

- *Identity:* Thể hiện tôi là ai? (Who Am I?). Để nhận dạng một người nào đó ta thường sử dụng tính chất đặc trưng duy nhất để mô tả chính mình.
- *Authentication (This is Who I am):* Khi cố gắng log in vào một web site, ta phải chuyển thông tin xác thực đi. Tiến trình của *authentication* chứng minh bạn đúng là người mà bạn đang nói đến. Có hai kiểu chiến lược xác thực Web authentication đó là: windows và forms authentication.
- *Authorization* (This is What I Can do): Authorization là bước tiếp theo của authentication. Khi người dùng chuyển username và password đến một web site, web server không chỉ kiểm tra password so khớp với username, mà còn xem xét các quyền (permissions) mà người dùng đó được trao bởi webmaster.

ĐĂNG NHẬP

- Ý nghĩa đăng nhập?
- Ý tưởng thiết kế?
  - Database: Thông tin người dùng, nhóm quyền, Người dùng thuộc nhóm quyền
  - Các Class: Account, AccountModel

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU LOGIN

- Vấn đề đơn giản nhất trong bảo mật website là yêu cầu người dùng Login. Ta có thể sử dụng Authorize action filter on a controller để thực hiện.
- Thuộc tính AuthorizeAttribute là authorization filter mặc định trong ASP.NET MVC.
  - Sử dụng nó để giới hạn truy cập đến một action method.
  - Thuộc tính này sử dụng như là shorthand áp dụng cho mọi action method trong controller.

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU LOGIN

```
// GET: /Admin/DanhMucAdmin/
[Authorize]
public ActionResult Index(){
      var dm = new DanhMucFunction().DanhMucs
                  .Where(p => p.TenDM != null);
      return View(dm);
}
```

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU LOGIN

- Bảo mật toàn bộ ứng dụng sử dụng Global Authorization Filter.
  - Toàn bộ application yêu cầu authorization:
    - Cấu hình AuthorizeAttribute như là global filter và
    - Cho phép anonymous truy cập đến controllers hoặc methods bằng cách sử dụng AllowAnonymous attribute.

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU LOGIN

- Bảo mật toàn bộ ứng dụng sử dụng Global Authorization Filter.
  - Để đăng ký AuthorizeAttribute như là global filter, ta đăng ký trong RegisterGlobalFilters method, ở \App_Start\FilterConfig.cs:

```
public class FilterConfig{
   public static void RegisterGlobalFilters
(GlobalFilterCollection filters){
   filters.Add(new System.Web.Mvc.AuthorizeAttribute());
            filters.Add(new HandleErrorAttribute());
    }
}
```

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU LOGIN

- Thuộc tính AllowAnonymous cho phép truy cập với người dùng Anonymous.

```
// GET: /Account/
[AllowAnonymous]
public ActionResult Login(string returnUrl){
    ViewBag.ReturnUrl = returnUrl;
    return View();
}
```

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU ROLE MEMBERSHIP

- Chỉ định một nhóm:

```
[Authorize(Roles="Administrator")]
public class StoreManagerController : Controller
```

- Chỉ định một số nhóm:

```
 [Authorize(Roles="Administrator,
SuperAdmin")]
 public class TopSecretController:Controller
```

SỬ DỤNG AUTHORIZE ATTRIBUTE ĐỂ YÊU CẦU ROLE MEMBERSHIP

- Chỉ định số người dùng:

```
[Authorize(Users="Jon,Phil,Scott,
Brad,David")]
public class TopSecretController:Controller
```

- Chỉ định nhóm và người dùng:

```
[Authorize(Roles="UsersNamedScott",
Users="Jon,Phil,Brad,David")]
public class TopSecretController:Controller
```

CUSTOM AUTHORIZE

- Xây dựng CustomPrincipal kế thừa IPrincipal:

```
public class CustomPrincipal:IPrincipal {
        private Account Account;
        public CustomPrincipal(Account account){
        this.Account = account;
            this.Identity = new
GenericIdentity(account.UserName);
        }
        public IIdentity Identity{ get;set;}
        public bool IsInRole(string role) {
            var roles = role.Split(new char[] { ',' });
            return roles.Any(r =>
this.Account.Roles.Contains(r));
       }
   }
```

CUSTOM AUTHORIZE

- Xây dựng CustomAuthorizeAttribute kế thừa AuthorizeAttribute:

```
public class CustomAuthorizeAttribute:AuthorizeAttribute{
    public override void OnAuthorization(AuthorizationContext
filterContext){
    if (string.IsNullOrEmpty(SessionPersister.UserName)){
       filterContext.Result = new RedirectToRouteResult( new
System.Web.Routing.RouteValueDictionary(new { Controller =
"Account", Action = "Index" })); }
    else { AccountModel am = new AccountModel();
CustomPrincipal cp = new
CustomPrincipal(am.Find(SessionPersister.UserName));
    if (!cp.IsInRole(Roles)){
         filterContext.Result = new RedirectToRouteResult(new
System.Web.Routing.RouteValueDictionary(new { Controller =
"Account", Action = "Index" })); } } } }
```

CUSTOM AUTHORIZE

- Xây dựng `SessionPersister` lấy và tạo Session lưu thông tin người đăng nhập.

```
public static class SessionPersister{
   static string usernameSessionvar = "username";
   public static string UserName{
      get{    if (HttpContext.Current == null){
                return string.Empty;}
                var sessionVar =
HttpContext.Current.Session[usernameSessionvar];
                if (sessionVar != null){
                    return sessionVar as string;
                }
                return null;}
        set {
          HttpContext.Current.Session[usernameSessionvar] = value;
            }
        }
    }
```

CUSTOM AUTHORIZE

- Chỉ định .

```
[CustomAuthorize(Roles="Admin")]
public ActionResult Index(){
    var dm = new DanhMucFunction().DanhMucs
                .Where(p => p.TenDM != null);
     return View(dm)
  }
```

THẢO LUẬN – CÂU HỎI