ДОДАТОК А

Лістинг програми

```
using System;
using System.Collections.Generic; using System.Linq;
using System.Runtime.Serialization; using
System.Security.Principal; using System.Text;
using System.Threading.Tasks;
using NetworkSecuritySystemm.InversionOfControl;
namespace NetworkSecuritySystemm
{
[DataContract] [Serializable]
public class Organization
{
private static readonly Lazy<IDataAccess> DataAccessLazy
= new Lazy<IDataAccess>(() => IoC.Resolve<IDataAccess>());
private static IDataAccess DataAccess => DataAccessLazy { get;
set;} [DataMember]
public string Id { get; set; }
[DataMember]
public string Name { get; set; }
[DataMember]
public string Settings { get; set; }
[DataMember]
public string CreationData { get; set; }
[DataMember]
public string ContactInfo { get; set; }
[DataMember]
public string License { get; set; }
public Task<IEnumerable<Account>> GetAllMembersAsync()
{
return await DataAccess.GetAllMembersAsync(Id);
}
public async Task RemoveAllMembersAsync()
{
return await DataAccess.RemoveAllMembersAsync(Id);
}
public async Task AddNewMemberAsync(Account account)
{
return await DataAccess.AddNewMemberAsync(account);
}
public async Task ResetSettingsToDefault()
{
return await DataAccess.ResetSettingsToDefaultAsync(Id);
}
public async Task<IEnumerable<Policy>> GetPoliciesAsync()
{
return await DataAccess.GetPoliciesAsync(Id);
}
}
}
```

```
using System; using System.Linq; using System.Net;
using System.Threading.Tasks; using System.Web;
using System.Web.Mvc;
using System.Web.Optimization; using System.Web.Routing;
using Microsoft.WindowsAzure.ServiceRuntime;
namespace NetworkSecuritySystemm
{
public class MvcApplication : BaseHttpApplication
{
protected override void Application_Start()
{
base.Application_Start();
BundleConfig.RegisterBundles(BundleTable.Bundles);
ModelBinders.Binders.DefaultBinder = new
JsonModelBinder(); BootstrapIoC();
ControllerBuilder.Current.SetControllerFactory(typeof(PortnoxCon
trollerFactory));
RouteConfig.RegisterRoutes(RouteTable.Routes);
GlobalFilters.Filters.Add(new
RequreHttpsForNonLocalRequestsFilter());
GlobalFilters.Filters.Add(new ValidateInputAttribute(false));
ViewEngines.Engines.OfType<RazorViewEngine>().FirstOrDefault()
.AddViewsRecursive(this.Server, "~/Views/App");
InitAsync().GetAwaiter().GetResult();
}
protected void Application_AcquireRequestState(object sender,
EventArgs e)
{
if (!new HttpRequestWrapper(Request).IsAjaxRequest() &&
HttpContext.Current.Session
!= null)
{
var httpCookie =
HttpContext.Current.Request.Cookies["timezoneoffset"]; if
(httpCookie != null)
{
Session["timezoneoffset"] = httpCookie.Value;
}
var languages = HttpContext.Current.Request.UserLanguages; if
(languages != null &&
languages.Length > 0)
{
Session[DateTimeUtils.UICultureKey] = languages[0];
}
}
}
#region Private methods
private void Application_Error(object sender, EventArgs e)
{
var ex = Server.GetLastError(); if (ex == null)
return;
var httpException = ex as HttpException; if (httpException !=
null)
```

```
{
var statusCode = (HttpStatusCode)httpException.GetHttpCode();
switch (statusCode)
{
case HttpStatusCode.NotFound: case HttpStatusCode.Forbidden:
goto SetResponseCode;
}
}
Logger.Trace (Errors.UnhandledError, ex); SetResponseCode:
Server.ClearError();
Response.StatusCode = httpException != null
? httpException.GetHttpCode()
: (int)HttpStatusCode.InternalServerError;
}
private async Task InitAsync()
{
var instance = await
RoleFunctionalitiesHolder.TryLoadInstanceAsync(RoleEnvironment.C
urrentRoleInstance.ShortI
d());
BaseController.SetRoleFunctionalitiesMode(instance);
}
private static void BootstrapIoC()
{
Bootstrapper.Initialise();
}
#endregion Private methods
private const string AssetsBasePath = "~/assets";
}
}
@using System.Configuration
@{
var sessionTimeout = FormsAuthConfig.Timeout.TotalMinutes; var
registrationModel = Model ??
new RegistrationViewModel();
}
<form id = "__AjaxAntiForgeryForm" action="#" method="post"
autocomplete="off">@Html.AntiForgeryToken()</form>@* this form
required for XSRF
protection*@
@{
Layout = !@User.Identity.IsAuthenticated ?
"~/Views/Shared/_NotAuthenticatedUserLayout.cshtml" :
"~/Views/Shared/_Layout.cshtml";
}
@if(@User.Identity.IsAuthenticated)
{
@Html.Partial("_Application")
}
else
{
<div class="row_login" style="display: none">
<!-- ko with: $data.loginModel().ViewModel -->
```

```
<div id = "eula-dialog" class="dialog-popup" style="display:
none" ">
<div class="inner">
<div class="scroll-content">
<div class="title">
<button class="btn-ico pull-right"><i class="ico"></i></button>
<label class="title"
style="">"Messages.Login_TermsAndConditionsTitle)</label>
</div>
</div>
</div>
</div>
<!-- /ko -->
<div class="enter_form enter_form_hidden">
<div id = "loginPanel" class="wrap_item" data-bind="with:
$data.loginViewModel().ViewModel" style="margin-top:25%;">
@Html.Partial("_Login", new
LoginViewModel())
<div id = "changePasswordPanel" class="wrap_item" data-
bind="visible: $data.Key ==
'ChangePassword'" style="display: none; margin-top: 25%">
@Html.Partial("_ChangePassword", new ChangePasswordViewModel())
<div class="clearfix"></div>
</div>
<div id = "forgotPasswordPanel" class="wrap_item" data-
bind="visible: $data.Key ==
'ForgotPassword'" style="display: none !important; margin-top:
25%">
@Html.Partial("_ForgotPassword", new ForgotPasswordViewModel())
<div class="clearfix"></div>
</div>
</div>
<div class="enter_form" data-bind="visible: $data.Key ==
'Registration'" style="display: none;">
<div class="scroll-wrapp">
<div class="header-register">
<div class="navigation-bar-content">
<a href = '@Url.Content("~/")' class="logotype" >
<img src = "@Url.Content("~/Content/images/logo.png")" >
</a>
</div>
</div>
<div id = "registerPanel" class="wrap_item">
@Html.Partial("_Register", (RegistrationModel)
model)
<div class="clearfix"></div>
</div>
</div>
</div>
<div class="img_wrap" data-bind="visible: $data.Key !=
'Registration'">
<div class="inner">
<h1 style = "display: none" data-bind="visible: $data.Key !=
```

```
'Registration'">Messages.Login_Title</h1>
<h1 style = "display: none" data-bind="visible: $data.Key ==
'Registration'">Messages.GettingStarted_Title</h1>
<div class="enter_text">
<div style = "display: none;" data-bind="visible: $data.Key !=
'Registration'">
<p>
"Messages.Login_GetStartedTitle)
</p>
<p>
<button id = "getStartedBtn" type="button" class="button
inverse" data- bind="click:
function(){ window.location = '@Url.Content("~/signup")';
}">"Messages.Login_GetStarted)</button>
</p>
</div>
<div style = "display: none;" data-bind="visible: $data.Key ==
'CheckYourInbox'">
<button type = "button" class="button inverse" data-bind="click:
function(){ window.location =
'@Url.Content("~/")'; }">"Messages.Login_SignIn)</button>
</div>
</div>
</div>
</div>
</div>
}
using System;
using System.Collections.Generic; using System.Linq;
using System.Runtime.Serialization; using
System.Security.Principal; using System.Text;
using System.Threading.Tasks;
using NetworkSecuritySystemm.InversionOfControl;
namespace NetworkSecuritySystemm
{
[DataContract] [Serializable] public class Device
{
private static readonly Lazy<IDataAccess> DataAccessLazy
= new Lazy<IDataAccess>(() => IoC.Resolve<IDataAccess>()); z
private static IDataAccess
DataAccess => DataAccessLazy { get; set;}
[DataMember]
public string Id { get; set; }
[DataMember]
public Account Account { get; set; }
[DataMember]
public Organization Org { get; set; }
[DataMember]
public DeviceType deviceType { get; set; }
[DataMember]
public OSType osType { get; set; }
[DataMember]
public IEnumerable<string> IpAddresses { get; set; }
```

```
[DataMember]
public DateTime CreationData { get; set; }
public Task<DeviceData> GetLastDataAsync()
{
return await DataAccess.GetLastDataAsync(Id);
}
public async Task<string> GetLastIpAddressAsync()
{
return await DataAccess.GetLastIpAddressAsync(Id);
}
public async Task<Account> GetAccountAsync()
{
return await DataAccess.GetAccountAsync();
}
public async Task SetStatusAsync()
{
return await DataAccess.SetDeviceStatusAsync(Id, Status);
}
}
}
using System;
using System.Collections.Generic; using System.Linq;
using System.Runtime.Serialization; using
System.Security.Principal;
using System.Text;
using System.Threading.Tasks;
using NetworkSecuritySystemm.InversionOfControl;
namespace NetworkSecuritySystemm
{
[DataContract] [Serializable]
public class DeviceData
{
private static readonly Lazy<IDataAccess> DataAccessLazy
= new Lazy<IDataAccess>(() => IoC.Resolve<IDataAccess>()); z
private static IDataAccess
DataAccess => DataAccessLazy { get; set;}
[DataMember]
public string Id { get; set; }
[DataMember]
public Device Device { get; set; }
[DataMember]
public IEnumerable<string> reportTimes { get; set; }
[DataMember]
public object geolocation { get; set; }
[DataMember]
public OSType applications { get; set; }
[DataMember]
public IEnumerable<string> openPorts { get; set; }
[DataMember]
public object passwordComplexity { get; set; }
public Task<DeviceData> GetLastReportTime()
{
return await DataAccess.GetLastReportTime(Id);
```

```
}
public async string GetDeviceId()
{
return Device.Id;
}
}
}
@using System.Runtime.Remoting.Messaging
@using System.Web.Optimization
<!DOCTYPE html>
<html>
<head>
<meta charset = "utf-8" >
< meta http-equiv= "X-UA-Compatible" content= "IE=edge" />
< link href= "@Url.Content("~/Content/images/favicon.ico")" rel=
"shortcut icon" type= "image/x-
icon" />
@Html.Portnox().RenderStyleAsset("main.css")
@Styles.Render("~/Content/applicationDesign")
@Styles.Render("~/css/jquery/customScroll")
<title>Messages.Header</title>
</head>
<body class="metro settings-page positions-fix"
id="documentBody">
@Html.Partial("_GoogleTagManager")
<div class="loader" id="globalLoader"><span></span></div>
<div class="popup_wrapp " id="errorContainer">
<div class="inner"></div>
<div class="content">
<h1 id = "popupTitle" ></ h1 >
< div class="div_p" id="popupText"></div>
<a class="button" title="Ok" id="popupConfirmButton">OK</a>
<a class="button" title="Cancel" style="display: none;"
id="popupCancel">@Html.FormatWithGlobalText(t =>
DisplayMessages.Cancel)</a>
</div>
</div>
<div class="flex-body">
@if(Request.IsAuthenticated)
{
< div class="main"> @RenderBody()
</div>
}
@if(Request.IsAuthenticated)
{
< header >
< div class="navigation-holder">
<nav class="navigation-bar light ">
<nav class="navigation-bar-content">
<a href = '@Url.Content("~/")' class="logotype" >
<img src = "/Content/images/logo.png">
</a> @Html.Partial("TabsBar")
@Html.Partial("NotificationPanel")
```

```
</nav>
</nav>
</div> @Html.Partial("Header")
</header>
}
</div>
</body>
</html>
```

ДОДАТОК Б

Демонстраційний матеріал

№ документа	Позначення	Найменування	Додаткові відомості
		Текстові документи	
1	ГЮИК 041929.034 ПЗ	Пояснювальна записка	А4, 70 с.
		Інші документи	
2		Лістинг програми	А4, 8 с.
3		Демонстраційний матеріал	А4, 18 с.

					ГЮИК 041929.034 ВД
Змін.	Арк	Номер докум.	Підп.	Дата	
Розробив		Фоменко А.Д.			Розроблення програмного модуля для кібербезпеки виробничого обладнання Відомість кваліфікаційної роботи
Перевірив		Максимова С.С.			
Норм.контр		Демська Н.П.			
Затвердив		Невлюдов І.Ш.			

Літер	Аркуш	Арк
Н	1	1

Кафедра КІТАР ХНУРЕ