**GDAPPFILES**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace GAppv3.Models

{

public class GDAppFiles

{

public string ID { get; set; }

public string FileName { get; set; }

public long? FileSize { get; set; }

public long? FileVersion { get; set; }

public DateTime? CreatedTime { get; set; }

public DateTime? ModifiedTime { get; set; }

}

}

**GDAPPREPO**

using Google.Apis.Auth.OAuth2;

using Google.Apis.Download;

using Google.Apis.Drive.v3;

using Google.Apis.Services;

using Google.Apis.Util.Store;

using System;

using System.Collections.Generic;

using System.IO;

using System.Threading;

using System.Web;

namespace GAppv3.Models

{

public class GDAppRepo

{

public static string[] Scopes = { DriveService.Scope.Drive };

public static DriveService GetService()

{

UserCredential authority;

using (var stream = new FileStream(@"D:\authorization\_token.json", FileMode.Open, FileAccess.Read))

{

String FolderPath = @"D:\";

String FilePath = Path.Combine(FolderPath, "GDriveServiceCreds.json");

authority = GoogleWebAuthorizationBroker.AuthorizeAsync(

GoogleClientSecrets.Load(stream).Secrets,

Scopes,

"user",

CancellationToken.None,

new FileDataStore(FilePath, true)).Result;

}

DriveService service = new DriveService(new BaseClientService.Initializer()

{

HttpClientInitializer = authority,

ApplicationName = "GAppv3",

});

return service;

}

public static List<GDAppFiles> GetDriveFiles()

{

DriveService service = GetService();

FilesResource.ListRequest FileListRequest = service.Files.List();

FileListRequest.Fields = "nextPageToken, files(id, name, size, version, createdTime, modifiedTime)";

IList<Google.Apis.Drive.v3.Data.File> files = FileListRequest.Execute().Files;

List<GDAppFiles> FileList = new List<GDAppFiles>();

if (files != null && files.Count > 0)

{

foreach (var file in files)

{

GDAppFiles File = new GDAppFiles

{

ID = file.Id,

FileName = file.Name,

FileSize = file.Size,

FileVersion = file.Version,

CreatedTime = file.CreatedTime,

ModifiedTime = file.ModifiedTime

};

FileList.Add(File);

}

}

return FileList;

}

public static void DeleteFile(GDAppFiles files)

{

DriveService service = GetService();

try

{

// Initial validation.

if (service == null)

throw new ArgumentNullException("service");

if (files == null)

throw new ArgumentNullException(files.ID);

// Make the request.

service.Files.Delete(files.ID).Execute();

}

catch (Exception ex)

{

throw new Exception("Request Files.Delete failed.", ex);

}

}

}

}

**HOMECONTROLLER**

using GAppv3.Models;

using System.IO;

using System.Web;

using System.Web.Mvc;

namespace GAppv3.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public ActionResult GetGoogleDriveFiles()

{

return View(GDAppRepo.GetDriveFiles());

}

[HttpPost]

public ActionResult DeleteFile(GDAppFiles file)

{

GDAppRepo.DeleteFile(file);

return RedirectToAction("GetGoogleDriveFiles");

}

}

}

**GETGOOGLEDRIVEFILES**

@model IEnumerable<GAppv3.Models.GDAppFiles>

@{

ViewBag.Title = "Software Security - OAuth Simple File Manager Application";

}

<h2>Software Security - OAuth Simple File Manager Application</h2>

<script src="https://code.jquery.com/jquery-3.4.0.min.js"></script>

<style type="text/css">

#header {

width: 100%;

background-color: #accae3;

text-align: center;

}

#layouttable {

border: 0px;

width: 100%;

font-family: 'Cocogoose Pro';

background-color: #d3edf0;

}

#layouttable td.col1 {

width: 20%;

vertical-align: top;

}

#layouttable td.col2 {

width: 60%;

vertical-align: top;

background-color: #c3d9db;

}

#layouttable td.col3 {

width: 20%;

vertical-align: top;

}

</style>

<center>

<table class="table" border="1">

<tr id="header">

<th>

@Html.DisplayNameFor(model => model.FileName)

</th>

<th>

@Html.DisplayNameFor(model => model.FileSize)

</th>

<th>

@Html.DisplayNameFor(model => model.FileVersion)

</th>

<th>

@Html.DisplayNameFor(model => model.CreatedTime)

</th>

<th>

@Html.DisplayNameFor(model => model.ModifiedTime)

</th>

<th>

Delete File

</th>

</tr>

@if (Model.Count() > 0)

{

foreach (var item in Model)

{

<tr id="layouttable">

<td>

@Html.DisplayFor(modelItem => item.FileName)

</td>

<td>

@{

long? KiloByte = @item.FileSize / 1024;

string NewSize = KiloByte + " KB";

}

@NewSize

</td>

<td>

@Html.DisplayFor(modelItem => item.FileVersion)

</td>

<td>

@string.Format("{0: MM/dd/yyyy}", Convert.ToDateTime(Html.DisplayFor(modelItem => item.CreatedTime).ToString()))

</td>

<td>

@string.Format("{0: MM/dd/yyyy}", Convert.ToDateTime(Html.DisplayFor(modelItem => item.ModifiedTime).ToString()))

</td>

<td>

@using (Html.BeginForm("DeleteFile", "Home", FormMethod.Post, new { enctype = "multipart/form-data" }))

{

<input type="hidden" name=Id value="@item.ID">

<input type="submit" class="DeleteFile" value="Delete" style="align-content:center" />

}

</td>

</tr>

}

}

else

{

<td colspan="6">The List is Empty</td>

}

</table>

</center>

<script>

$(document).on('click', '.DownloadFile', function () {

debugger;

var fileId = $(this).attr("data-key");

window.location.href = '/Home/DownloadFile/' + fileId;

});

</script>

**ROUTECONFIG**

@model IEnumerable<GAppv3.Models.GDAppFiles>

@{

ViewBag.Title = "Software Security - OAuth Simple File Manager Application";

}

<h2>Software Security - OAuth Simple File Manager Application</h2>

<script src="https://code.jquery.com/jquery-3.4.0.min.js"></script>

<style type="text/css">

#header {

width: 100%;

background-color: #accae3;

text-align: center;

}

#layouttable {

border: 0px;

width: 100%;

font-family: 'Cocogoose Pro';

background-color: #d3edf0;

}

#layouttable td.col1 {

width: 20%;

vertical-align: top;

}

#layouttable td.col2 {

width: 60%;

vertical-align: top;

background-color: #c3d9db;

}

#layouttable td.col3 {

width: 20%;

vertical-align: top;

}

</style>

<center>

<table class="table" border="1">

<tr id="header">

<th>

@Html.DisplayNameFor(model => model.FileName)

</th>

<th>

@Html.DisplayNameFor(model => model.FileSize)

</th>

<th>

@Html.DisplayNameFor(model => model.FileVersion)

</th>

<th>

@Html.DisplayNameFor(model => model.CreatedTime)

</th>

<th>

@Html.DisplayNameFor(model => model.ModifiedTime)

</th>

<th>

Delete File

</th>

</tr>

@if (Model.Count() > 0)

{

foreach (var item in Model)

{

<tr id="layouttable">

<td>

@Html.DisplayFor(modelItem => item.FileName)

</td>

<td>

@{

long? KiloByte = @item.FileSize / 1024;

string NewSize = KiloByte + " KB";

}

@NewSize

</td>

<td>

@Html.DisplayFor(modelItem => item.FileVersion)

</td>

<td>

@string.Format("{0: MM/dd/yyyy}", Convert.ToDateTime(Html.DisplayFor(modelItem => item.CreatedTime).ToString()))

</td>

<td>

@string.Format("{0: MM/dd/yyyy}", Convert.ToDateTime(Html.DisplayFor(modelItem => item.ModifiedTime).ToString()))

</td>

<td>

@using (Html.BeginForm("DeleteFile", "Home", FormMethod.Post, new { enctype = "multipart/form-data" }))

{

<input type="hidden" name=Id value="@item.ID">

<input type="submit" class="DeleteFile" value="Delete" style="align-content:center" />

}

</td>

</tr>

}

}

else

{

<td colspan="6">The List is Empty</td>

}

</table>

</center>

<script>

$(document).on('click', '.DownloadFile', function () {

debugger;

var fileId = $(this).attr("data-key");

window.location.href = '/Home/DownloadFile/' + fileId;

});

</script>