

# Lab Notes: Investigating File Integrity in Microsoft OneDrive Using the Graph API

## Setup:

### Environment:

- OS: Windows 11
- .NET version: 8.0.403
- IDE: Visual Studio 2022

### Initial Setup:

- Installed NuGet packages using “dotnet add package {package\_name}” command: Azure.Identity (1.15.0), Microsoft.Graph(5.91.0) Microsoft.Extensions.Configuration.Json(9.0.8)
- Created Personal Microsoft Account
- Created an App registration for the Application in Azure Portal->Azure Active Directory (Figure 1) and Added Permissions(User.Read and Files.ReadWrite) for Microsoft Graph as delegated permissions.(Figure 2)

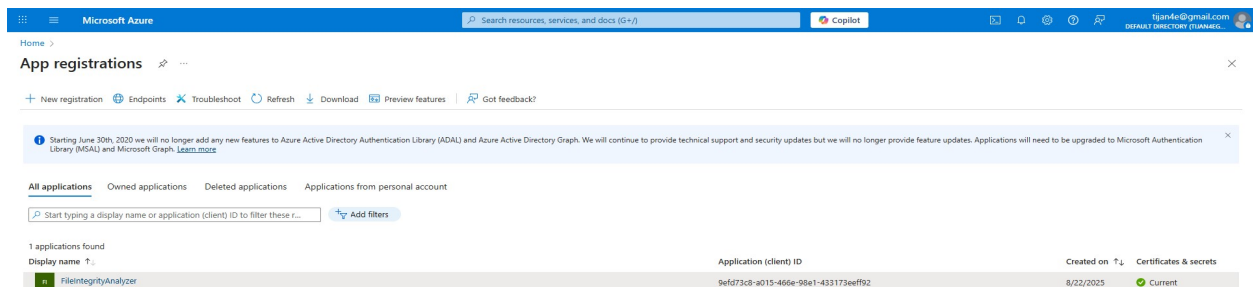


Figure 1 Creating App Registration

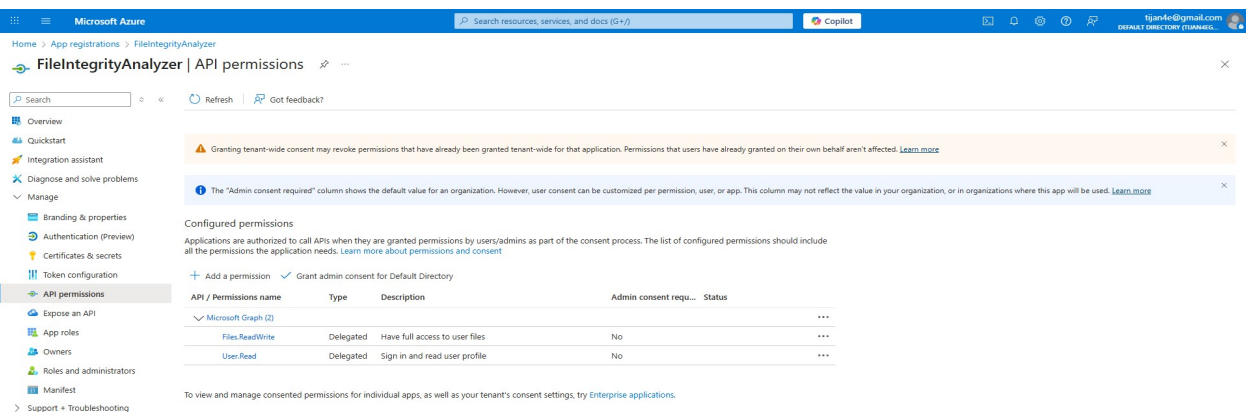


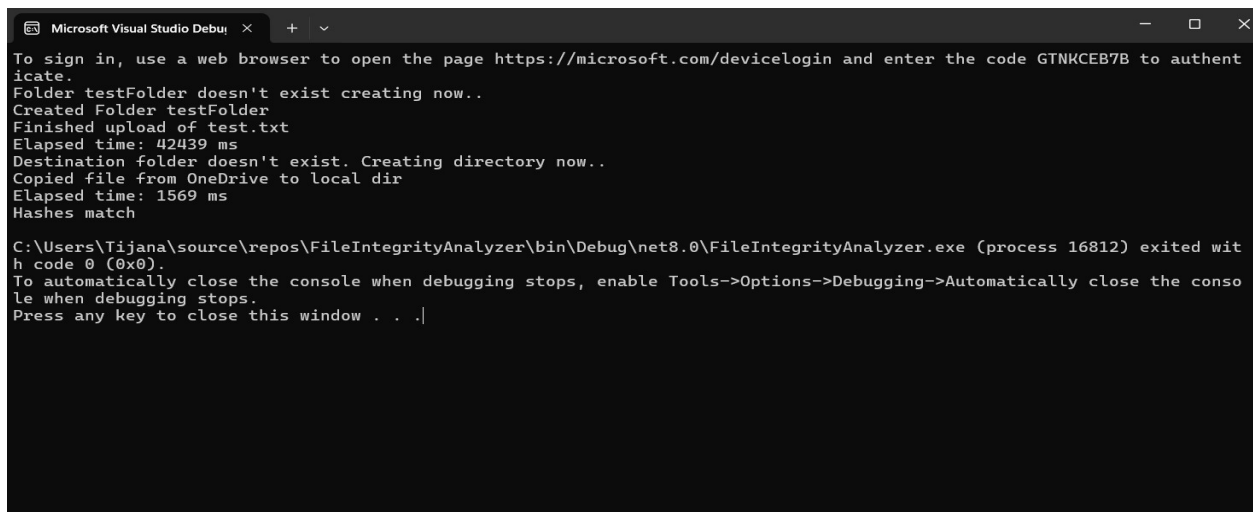
Figure 2 Adding Permissions for Microsoft Graph API

## Project Setup:

- Main entry point - Program.cs – for analysis – checks if hashes of file that is uploaded and file that is downloaded from OneDrive match.
- Chosen to divide the workflow into different classes
  1. Authentication – handles authentication using device code flow to Microsoft Graph and returns a GraphServiceClient.
  2. Uploader – handles uploading file to folder in signed-in user's OneDrive.
  3. Downloader – handles downloading file from signed-in user's OneDrive to a local directory.
  4. IntegrityVerifier – handles computing the SHA256 hash of a file and compares files' hashes.
- Config file appsettings.json that holds values for files' paths.

## Experiments and results:

- Experiment1: Upload and download small text test.txt file and check hashes(Figure3):
  1. File size: 394 B
  2. First time creating folder in OneDrive
  3. Upload time: 42439 ms
  4. First time creating local destination folder
  5. Download time: 1569 ms
  6. Hashes match: TRUE



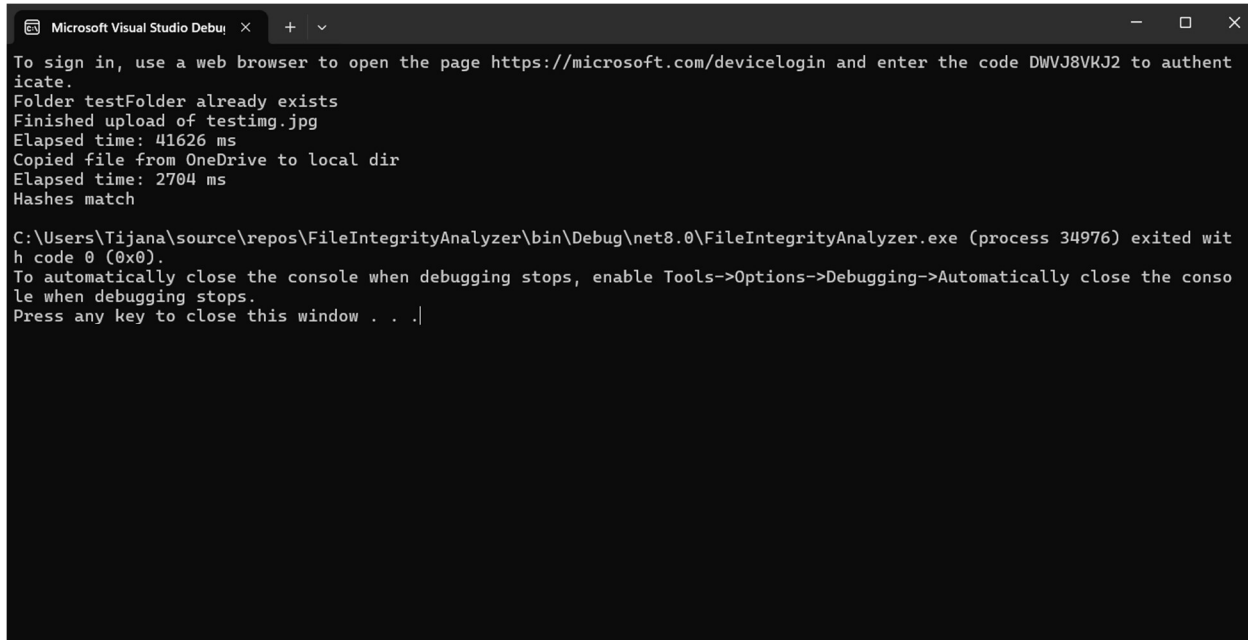
```
Microsoft Visual Studio Debug Console
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code GTNKCEB7B to authenticate.
Folder testFolder doesn't exist creating now..
Created Folder testFolder
Finished upload of test.txt
Elapsed time: 42439 ms
Destination folder doesn't exist. Creating directory now..
Copied file from OneDrive to local dir
Elapsed time: 1569 ms
Hashes match

C:\Users\Tijana\source\repos\FileIntegrityAnalyzer\bin\Debug\net8.0\FileIntegrityAnalyzer.exe (process 16812) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Figure 3 Results for test.txt file

- Experiment2: Upload and download of image testing.jpg file and check hashes(Figure4):
  1. File size: 5.27 MB
  2. Folder in OneDrive already exists
  3. Upload time: 41626 ms

4. Destination folder already created
5. Download time: 2704 ms
6. Hashes match: TRUE



```

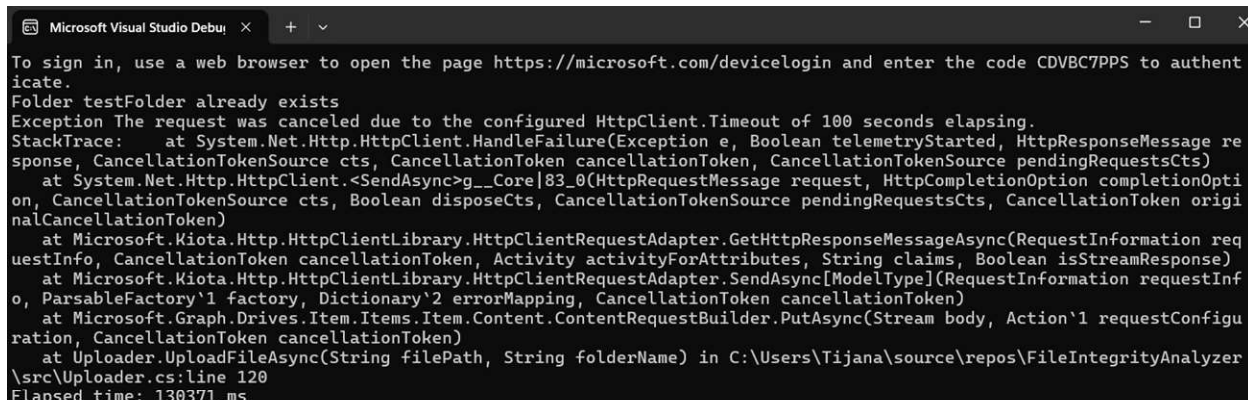
Microsoft Visual Studio Debug Console
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code DWVJ8VKJ2 to authenticate.
Folder testFolder already exists
Finished upload of testimg.jpg
Elapsed time: 41626 ms
Copied file from OneDrive to local dir
Elapsed time: 2704 ms
Hashes match

C:\Users\Tijana\source\repos\FileIntegrityAnalyzer\bin\Debug\net8.0\FileIntegrityAnalyzer.exe (process 34976) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|

```

Figure 4 Results for testimg.jpg file

- Experiment3: Upload and download of video testvid.mp4 file and check hashes(Figure5)
  1. File size: 121 MB
  2. Folder in OneDrive already exists
  3. ERROR: "Exception The request was canceled due to the configured HttpClient.Timeout of 100 seconds elapsing." This could be due to slow connection or the size of the file.



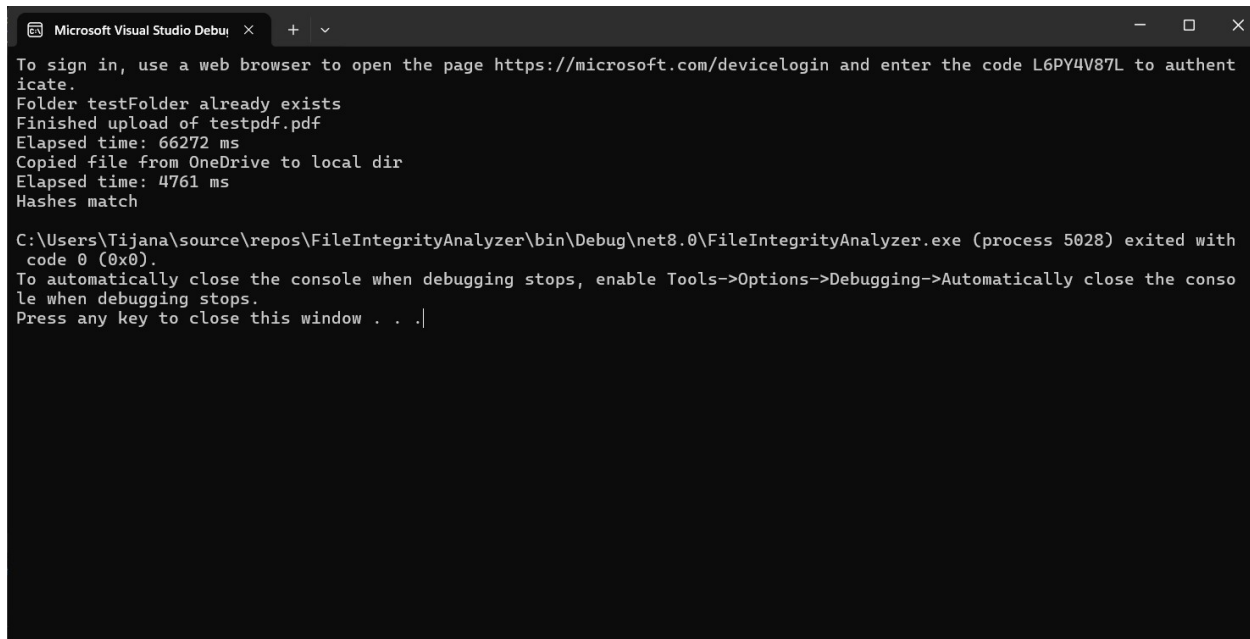
```

Microsoft Visual Studio Debug Console
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code CDVBC7PPS to authenticate.
Folder testFolder already exists
Exception The request was canceled due to the configured HttpClient.Timeout of 100 seconds elapsing.
StackTrace: at System.Net.Http.HttpClient.HandleFailure(Exception e, Boolean telemetryStarted, HttpResponseMessage response, CancellationTokenSource cts, CancellationToken cancellationToken, CancellationTokenSource pendingRequestsCts)
at System.Net.Http.HttpClient.<SendAsync>g__Core|83_0(HttpRequestMessage request, HttpCompletionOption completionOption, CancellationTokenSource cts, Boolean disposeCts, CancellationTokenSource pendingRequestsCts, CancellationToken originalCancellationToken)
at Microsoft.Kiota.Http.HttpClientLibrary.HttpClientRequestAdapter.GetHttpResponseMessageAsync(RequestInformation requestInfo, CancellationToken cancellationToken, Activity activityForAttributes, String claims, Boolean isStreamResponse)
at Microsoft.Kiota.Http.HttpClientLibrary.HttpClientRequestAdapter.SendAsync[ModelType](RequestInformation requestInfo, ParsableFactory`1 factory, Dictionary`2 errorMapping, CancellationToken cancellationToken)
at Microsoft.Graph.Drives.Item.Items.Item.Content.ContentRequestBuilder.PutAsync(Stream body, Action`1 requestConfiguration, CancellationToken cancellationToken)
at Uploader.UploadFileAsync(String filePath, String folderName) in C:\Users\Tijana\source\repos\FileIntegrityAnalyzer\src\Uploader.cs:line 120
Elapsed time: 130371 ms

```

Figure 5 Results for testvid.mp4

- Experiment4: Upload and download of pdf testpdf.pdf file and check hashes(Figure6):
  1. File size: 15.6 MB
  2. Folder in OneDrive already exists
  3. Upload time: 66272 ms
  4. Destination folder already exists
  5. Download time: 4761 ms
  6. Hashes match: TRUE



```
Microsoft Visual Studio Debug Console
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code L6PY4V87L to authenticate.
Folder testFolder already exists
Finished upload of testpdf.pdf
Elapsed time: 66272 ms
Copied file from OneDrive to local dir
Elapsed time: 4761 ms
Hashes match

C:\Users\Tijana\source\repos\FileIntegrityAnalyzer\bin\Debug\net8.0\FileIntegrityAnalyzer.exe (process 5028) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

*Figure 6 Results for testpdf.pdf*

## Errors and challenges:

- Encountered an error when first authenticating because the Allow public client flows was disabled in the Authentication part of the App Registration. By enabling public client flows the app can authenticate the user directly via device code or interactive window without the use of a client secret.
- Timeout error when trying to upload a larger file (reference Experiment4) because the HttpClient.Timeout is 100 seconds. This can be fixed by increasing the timeout on the HttpClient.

**Conclusions:**

- The biggest takeaway is that OneDrive preserves data integrity meaning the file did not change during the upload and download process.
- Another takeaway is that the time taken to upload and download the files depends on the file size, as well as network speed and number of api requests(like to create a folder in OneDrive to store the file) before the actual api request to upload/download.