

PDF sync with SharePoint Library

Setting-up One Drive Sync to the SharePoint site

- Sing-in to your Office 365 account.

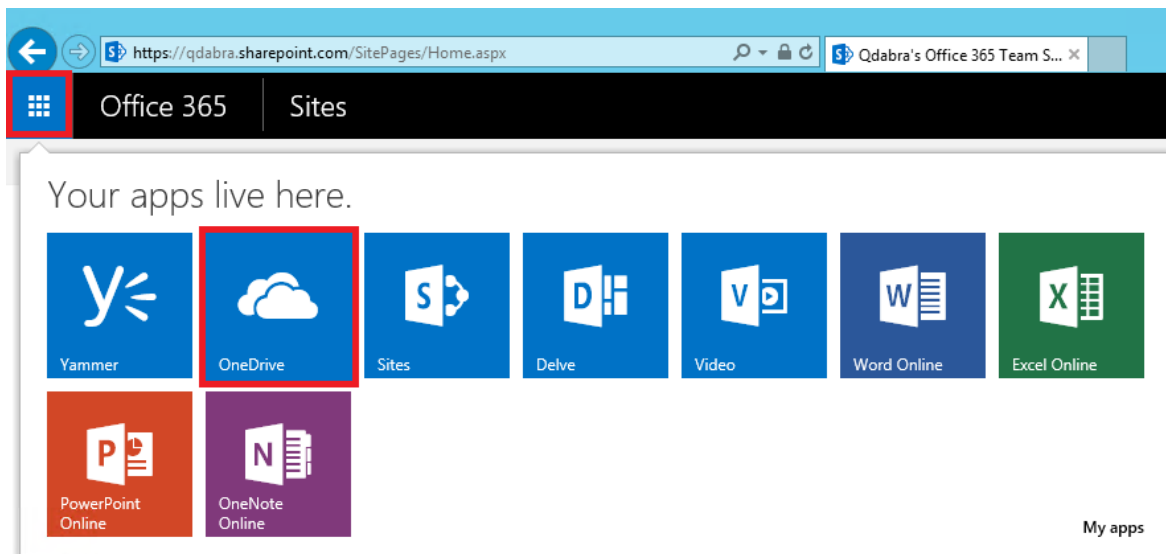


Sign in with your work or school account

Keep me signed in

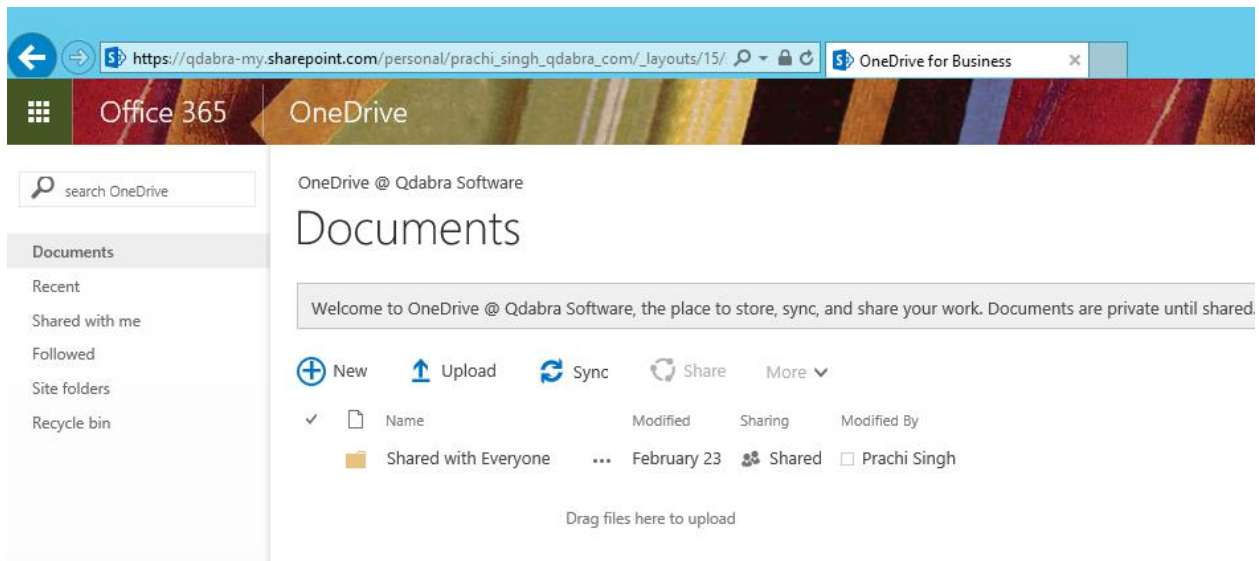
Sign in

- Click on the upper left corner grid on your SharePoint site.
- Click on **OneDrive** icon.

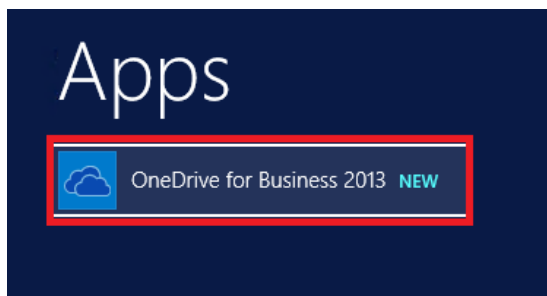


The screenshot shows a web browser window with the URL <https://qdabra.sharepoint.com/SitePages/Home.aspx>. The page header includes the Office 365 logo and the word "Sites". Below the header, the text "Your apps live here." is displayed. A grid of application icons is shown, including Yammer, OneDrive, Sites, Delve, Video, Word Online, Excel Online, PowerPoint Online, and OneNote Online. The OneDrive icon is highlighted with a red border. A "My apps" link is visible in the bottom right corner.

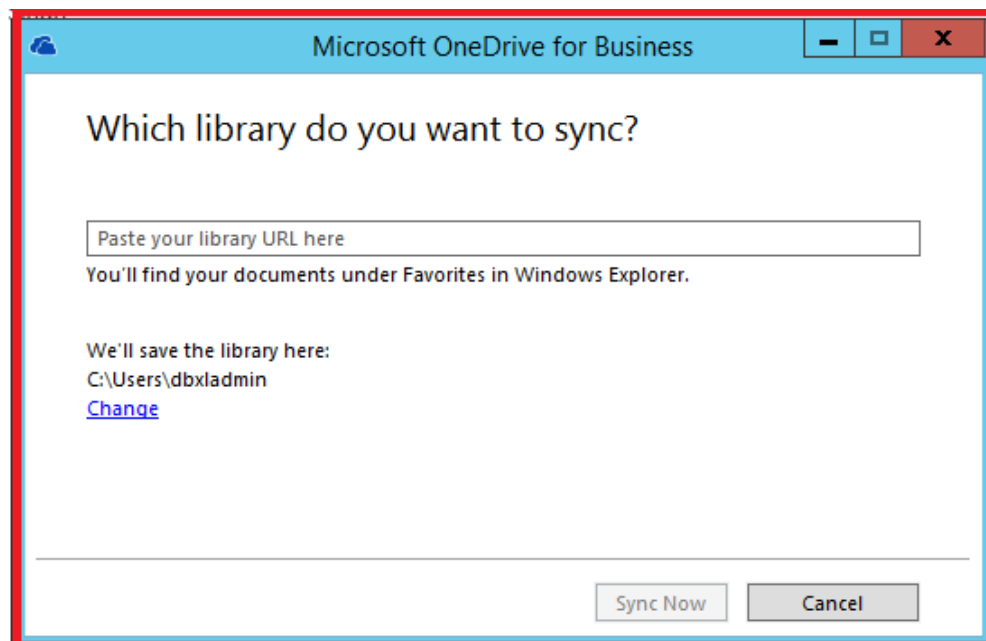
- This will open up OneDrive from your account.



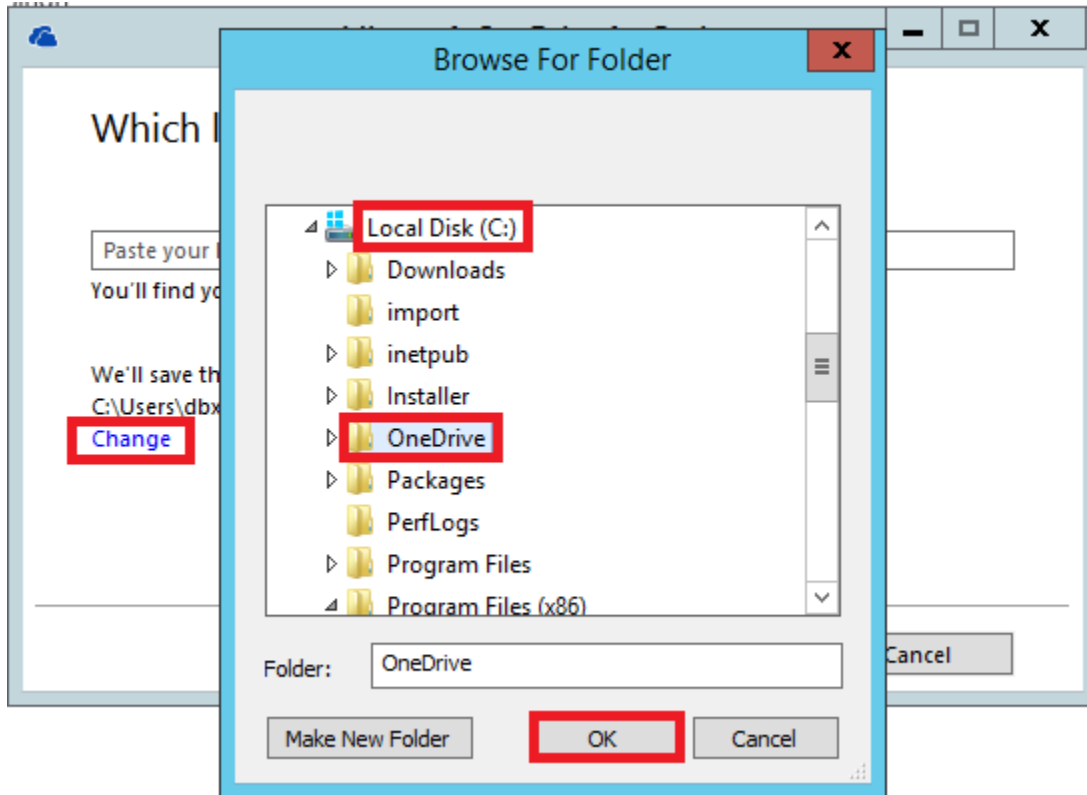
- Search for **One Drive for Business** on your system and open it.



- A “Microsoft OneDrive for Business” window appears.



- We are setting this up to share PDF's in a folder other than your O365 account. So, you need to create "**OneDrive**" folder under C: to sync and share PDF's.
 - Click on **Change** and navigate to the **OneDrive** folder under C:\.
 - Click **OK**.



- Now, you need to give the URL of the SharePoint library (Document Library) where you want to sync and share the PDFs.
 - Create a Document Library on your SharePoint site to sync the PDF's, if you haven't created it already.
 - On the SharePoint site, click on *Site Contents* and then click on **add an app** icon.



Site Contents

- Home
- Notebook
- Documents
- Apps in Testing
- Samples
- Developer Center
- Recent
 - PDF
 - PictureXMLTemplate
 - PicturePerfect
 - Signature Admin
 - PIN List
 - Site Contents**
 - Recycle Bin

Lists, Libraries, and other Apps

	add an app		App Packages 0 items Modified 4 weeks ago
	Form Templates 0 items Modified 4 weeks ago		MicroFeed 2 items Modified 4 months ago
	PIN List 30 items Modified 2 weeks ago		Signature Admin 0 items Modified 6 days ago

- Select **Document Library** from the *Apps You Can Add*.



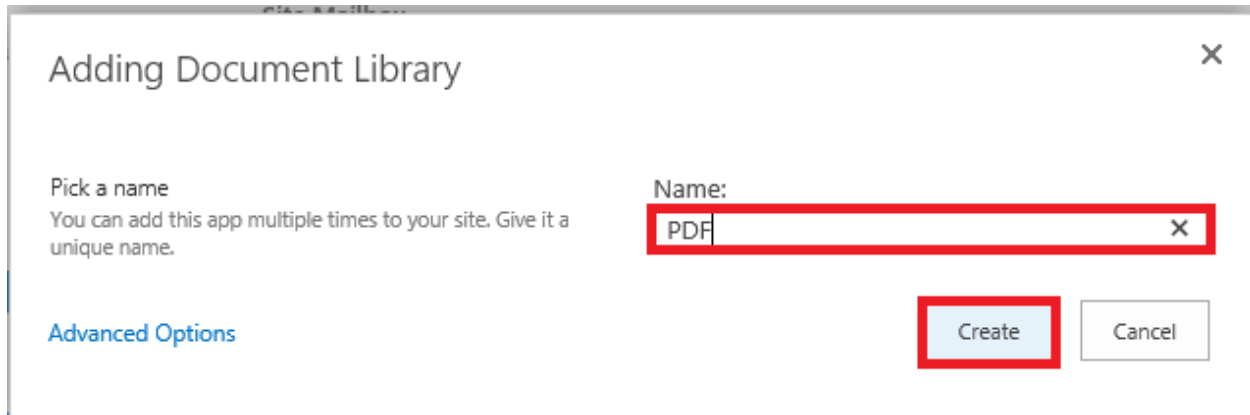
Site Contents › Your Apps

- Your Apps
- Apps You Can Add**
- From Your Organization
- Manage Licenses
- Your Requests
- SharePoint Store

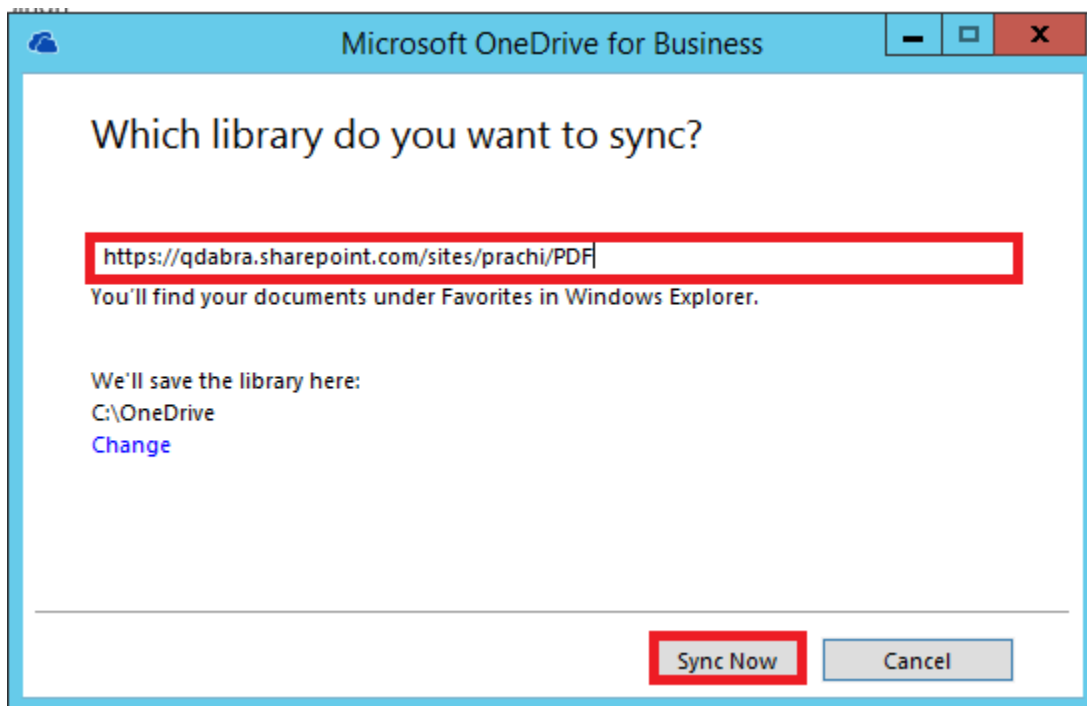
Find an app

Noteworthy			
Document Library Popular built-in app App Details	Custom List Popular built-in app App Details	Tasks Popular built-in app App Details	Site Mailbox Popular built-in app App Details

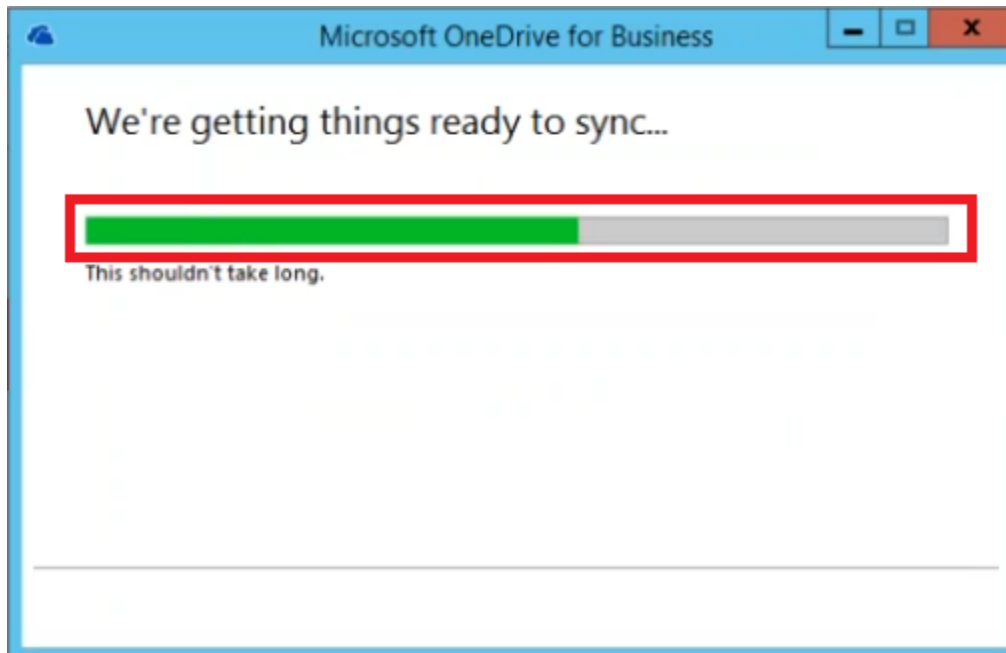
- Enter the name of document library (e.g. PDF) where you want to sync the PDF's.
- Click on **Create**.



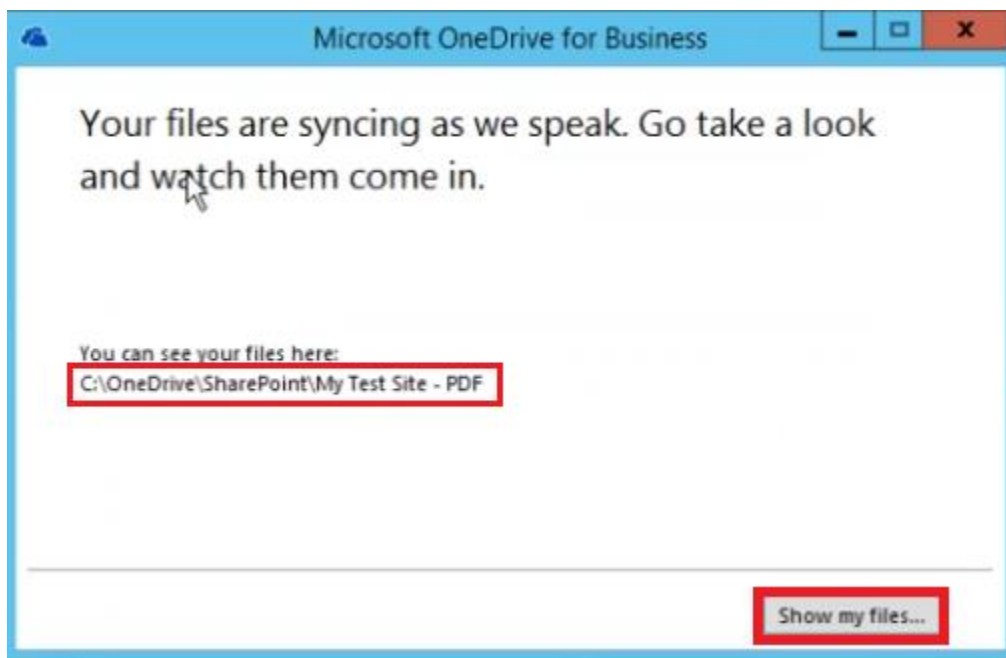
- Go back to the “**Microsoft OneDrive for Business**” window and enter the URL to the Document Library that you created to store PDFs.
- Click on **Sync now**.



- You will see a window showing a progress bar while it gets ready to sync the files.



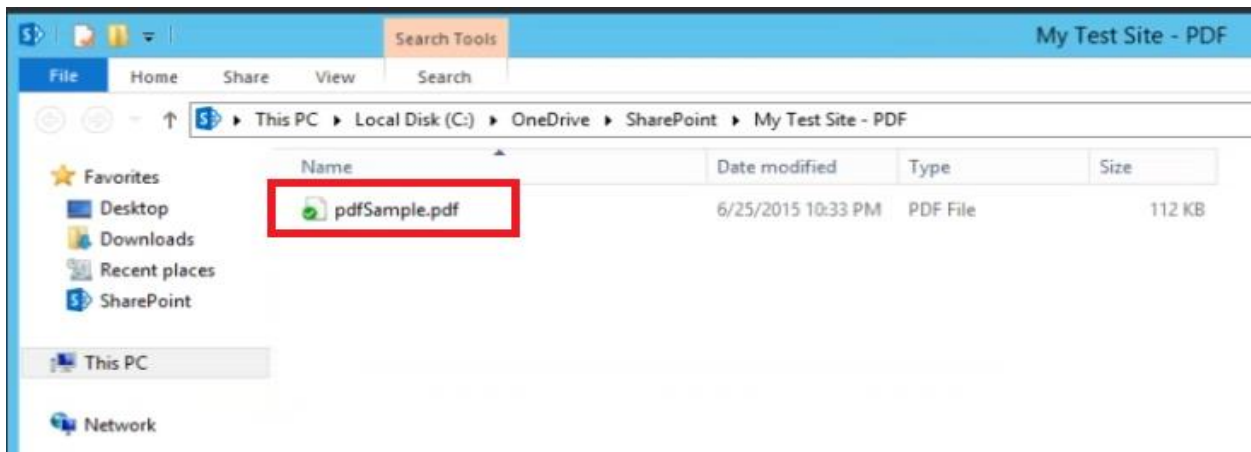
- Once the sync starts, you can go and see the files getting synced. Click on **Show my files...** button.



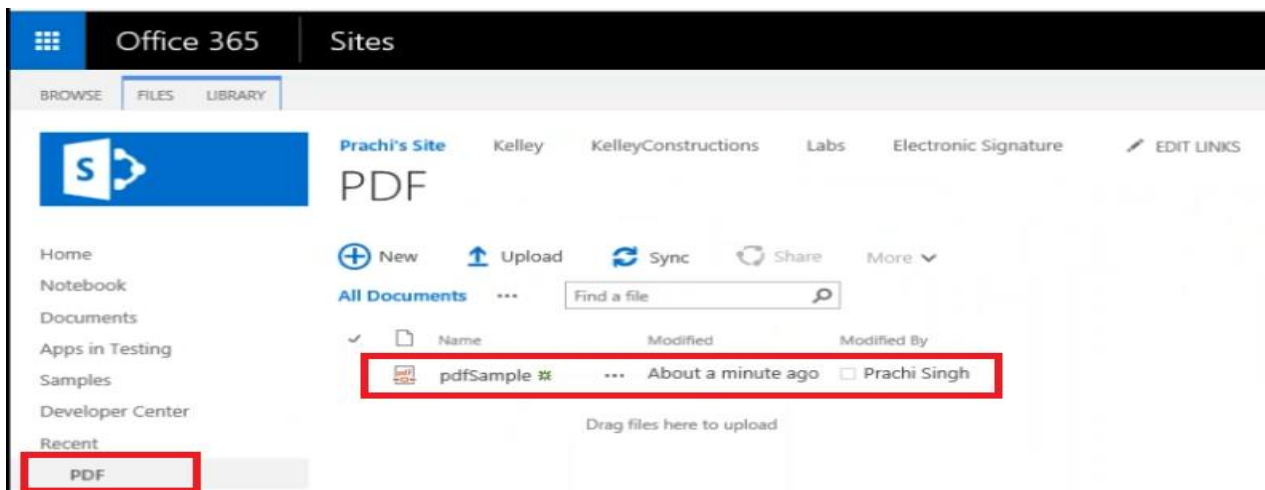
- It creates a **SharePoint** folder underneath the **OneDrive** folder in **C: that you specified earlier**. And under SharePoint folder. It creates another folder for the Library that you are synchronizing (**My Test Site – PDF**).
Note: For each library that you want to synchronize, you will have to configure it separately.



- Initially the folder is empty because it doesn't have any files yet.
- To test that the PDF's will sync with the SharePoint library, you can copy a sample PDF to your SharePoint folder.

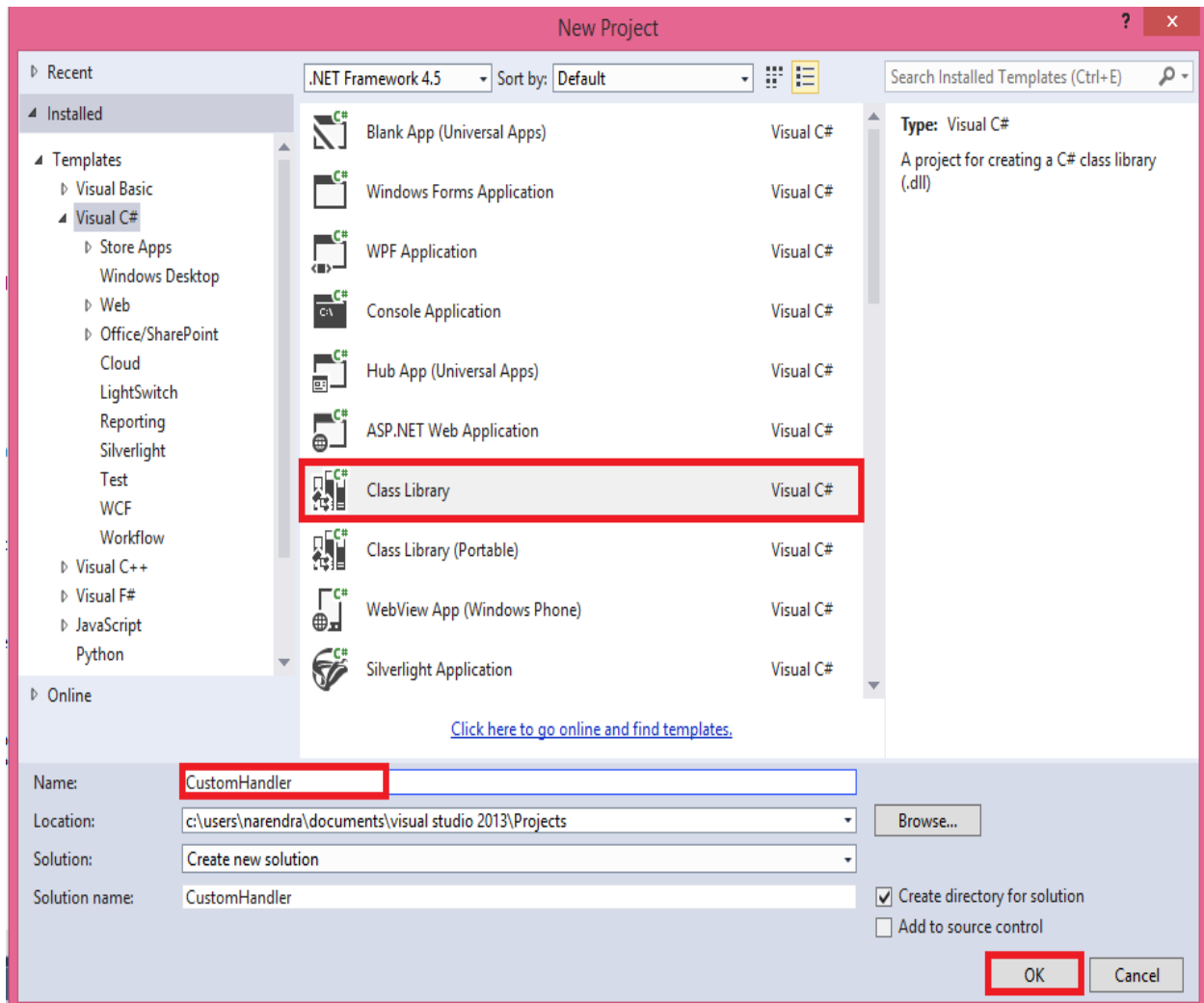


- Go back to your SharePoint library and check if the PDF sync was successful.
- If the sync was successful, you will see the PDF you just copied to the OneDrive Folder, on your SharePoint library.



Configure Custom dll to write PDFs to the SharePoint folder

- Now, we need to configure the custom dll to write the PDFs to the SharePoint folder that synchronizes with your SharePoint library.
- For this we need to create a custom dll on Visual Studio.
 - Open Visual Studio 2013 and create a new **Class Library** project, name it *CustomHandler*.
 - Click OK



- Here is the code that is used for the PDF sync to the SharePoint library.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Xml;
using System.Xml.XPath;
using System.IO;

namespace CustomHandler
{
    1 reference
    public class CustomRendering
    {
        public const string c_LogSource = "Qdabra DBXL Rendering";
        private string m_outputFolder;
        private string m_filenameXPath;

        0 references
        public CustomRendering()
        {
            m_outputFolder = System.Configuration.ConfigurationManager.AppSettings["PdfOutputFolder"];
        }

        0 references
        public int RenderingCompleted(int docid, string documentXml, byte[] pdf)
        {
            try
            {
                string baseName = GetFilename(docid, documentXml);
                string filename = string.Format("{0}.pdf", baseName);
                string pdfPath;

                pdfPath = System.IO.Path.Combine(m_outputFolder, filename);
                string dirname = Path.GetDirectoryName(pdfPath);
                if (!Directory.Exists(dirname))
                {
                    Directory.CreateDirectory(dirname);
                }
                File.WriteAllBytes(pdfPath, pdf);
            }
        }
    }
}

```

```

        // Add record to metadata table
    }
    catch (Exception ex)
    {
        System.Diagnostics.EventLog.WriteEntry(c_LogSource, "Error processing PDF: " + ex.ToString());
        return -1;
    }

    return 0;
}

1 reference
public string GetFilename(int docid, string xml)
{
    using (System.IO.StringReader reader = new System.IO.StringReader(xml))
    {
        XPathDocument xdoc = new XPathDocument(reader);
        XPathNavigator nav = xdoc.CreateNavigator();
        string wonumber = GetValue(nav, "//*[local-name()='WONumb']");
        string formName = GetValue(nav, "//*[local-name()='FormName']");
        return string.Format("{0} {1}", wonumber, formName);
    }
}

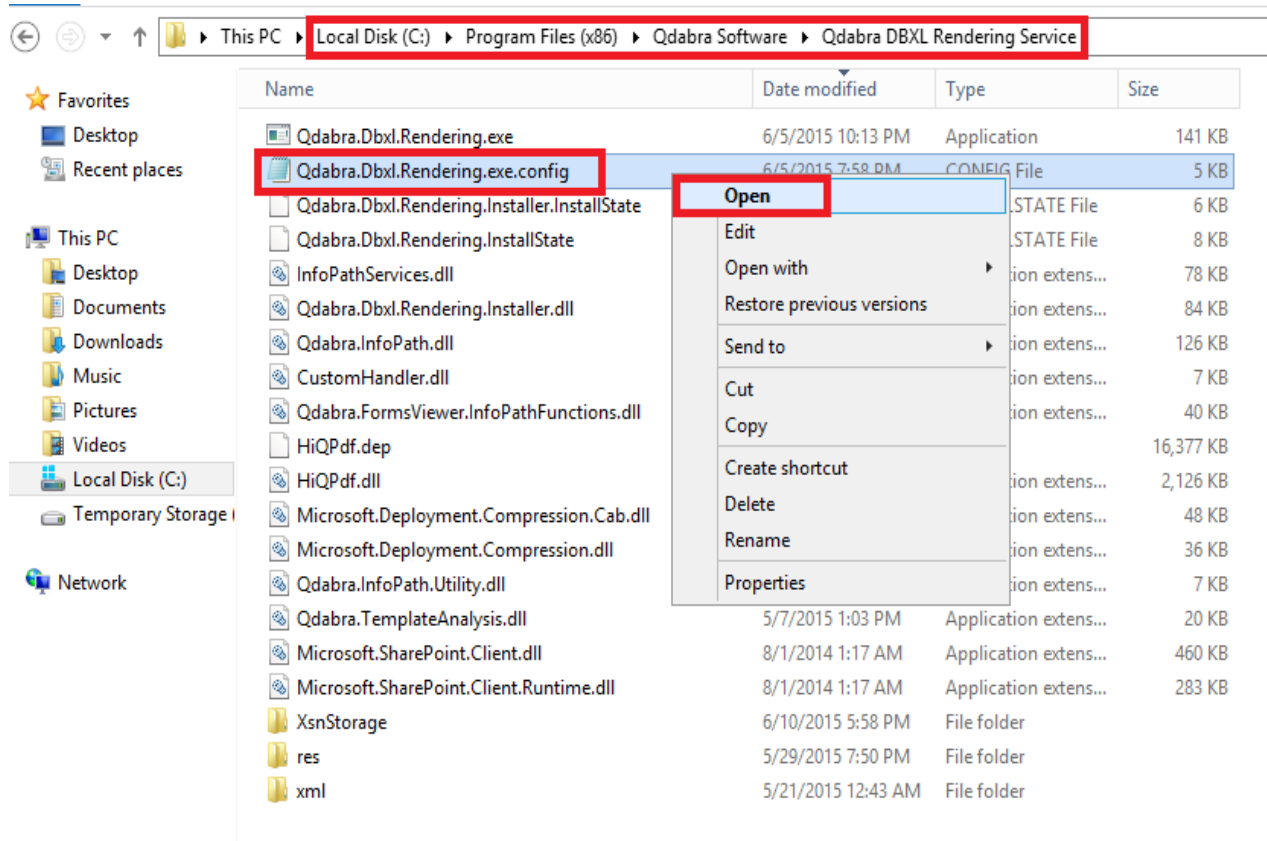
2 references
private string GetValue(XPathNavigator nav, string xpath)
{
    nav = nav.SelectSingleNode(xpath);
    if (nav != null)
    {
        return nav.Value;
    }
    return "";
}
}
}

```

- The **CustomHandler.dll** looks into the **Qdabra.Dbxl.Rendering.exe.config** file for the location to write the pdf to.

Change the **Qdabra.Dbxl.Rendering.exe.config** file to write PDFs to the SharePoint folder

- Navigate to **C:/ProgramFiles(x86)/QdabraSoftware/QdabraDbxlRenderingService** on your system.
- Right-click on **Qdabra.Dbxl.Rendering.exe.config** file and open with Notepad.



- Search for Custom Handler, this will be called after Pdf rendering. It looks like this:

<add key="CustomHandler" value="CustomHandler.CustomRendering, LedcorPdfSync" />

Here the syntax is:

"CustomHandler" value ="Namespace.ClassName, AssemblyName"

- Customize the value of *CustomHandler* according to the namespace, class name and Assembly name of your dll.
- Give the PdfOutputFolder value i.e. the path to the Pdf folder synchronizing with OneDrive.

```
Qdabra.Dbxl.Rendering.exe.config - Notepad
File Edit Format View Help

-->

<add key="Smtp_Servername" value="smtp.sendgrid.net" />
<add key="Smtp_Port" value="25" />
<add key="Smtp_UserName" value="azure_404d9f83db1b6fa77ba28f16421128c3@azure.com" />
<add key="Smtp_SequenceNumber" value="" />
<add key="Smtp_Password" value="TWU0zm5lip8Jy9d" />
<add key="Smtp_EnableSsl" value="true" />

<add key="Email_Subject" value="" />
<add key="Email_Body" value="" />
<add key="Email_BodyIsHtml" value="false" />
<add key="Email_FromAddress" value="Notifications@formsboard.com" />
<add key="Email_FromDisplayName" value="DBXL Rendering Service" />
<add key="Email_FromLocked" value="false" />

<!-- CustomHandler that will be called after rendering -->
<add key="CustomHandler" value="CustomHandler.CustomRendering, LedcorPdfSync" />

<!-- Properties sepcific to the CustomHandler implementation -->
<add key="PdfMetadataConnString" value="" />
<add key="PdfOutputFolder" value="C:/OneDrive/SharePoint/MyTestSite-PDF" />
<add key="PdfFileNameXPath" value="" />

</appSettings>

<system.serviceModel>
  <bindings>
    <basicHttpBinding>
      <binding name="IDbxlDocumentService" maxBufferSize="2147483647" maxReceivedMessageSize="2147483647"
        <security mode="Transport">
          <transport clientCredentialType="Ntlm" proxyCredentialType="None" realm="" />
          <message clientCredentialType="UserName" algorithmSuite="Default" />
        </security>
      </binding>
      <binding name="IQueryDBService" maxBufferSize="2147483647" maxReceivedMessageSize="2147483647">
        <security mode="Transport">
          <transport clientCredentialType="Ntlm" proxyCredentialType="None" realm="" />
          <message clientCredentialType="UserName" algorithmSuite="Default" />
        </security>
      </binding>
    </bindings>
  </system.serviceModel>
```

- Save the **Qdabra.Dbxl.Rendering.exe.config** file.
- Add the **CustomHandler.dll** to the `C:/ProgramFiles(x86)/QdabraSoftware/QdabraDbxlRenderingService` folder.
- Test it by submitting a form and generating a PDF of it. Then go to your SharePoint library and look for the new PDF.