**Local project folder preparation:**

After creating the folder ‘AdhocEDI’ according the same folder structure as in GitLab, you have to create a Git repository, by right-clicking the AdhocEDI folder and selecting ‘Git Create repository here..’

Graphical user interface, text, application, email

Description automatically generated

In the AdhocEDI folder create a readme.md file (copy a previous one and alter the contents).

# AdhocEDI

Webservice that acts as an interface between the (old) Adhoc customers and SAP:

Type: Service with webserver on port 8080

Location: \\Dealer\c$\services

Maintainer: Wilfred Geerlings

Info:

AdhocEDI.dproj: Project file for creating the ISAPI DLL

AdhocEDI\_APPL.dproj: Project file for the standalone testing and debugging application

AdhocEDI\_ProjectGroup.groupproj: Project group for both projects

Put a <space> after every line when there is no empty line after it, otherwise it will not put an <CrLf> after the line.

Now copy a ‘.gitignore’ file from a previous Delphi-Git project in this folder too. It makes sure that all irrelevant files will not show up in SourceTree.

Create a ‘Delphi\_10’ folder, here is where you will put your Delphi\_10 sources. We will probably need a ‘Docs’ folder too. Tip: rename your old folder by adding ‘ moved to RAD’, that way the IDE will not take wrong files.

Graphical user interface, application, Word

Description automatically generated

**Create a new Project in GitLab:**

Select the right group first, in this case it is the: [eCommerce Handler](https://gitlab.com/apollotyres/avbv/rad/ecommerce-handler)

Select New ‘project’.

A picture containing chart

Description automatically generated

Provide the project name, ‘AdhocEDI’

Graphical user interface, text, application, Word

Description automatically generated

Enter a short project description. This line could be the start of your Read.me file too.

Make sure to uncheck the checkbox: Initialize repository with a README!

**In SourceTree we will open the generated Git repository:**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, table

Description automatically generated

You have to add the remote GitLab project you have created previously:

The needed URL can be found in GitLab:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Check ‘Default remote’ and keep ‘Generic Account’ as Remote account, at the moment Source Tree does not work properly when you select GitLab as Remote Account.

The remote button will now automatically open the correct project in GitLab, when pressed!

Graphical user interface, application

Description automatically generated with medium confidence

**Your repository is still empty, so you have to add your source now 😊**

When copying from a previous folder, make sure you rename the ‘old’ folder so that you know which folder to use and prevents ‘open’ file in your project to be linked to the ‘old’ folder without you noticing. It will act as a backup too.

Graphical user interface, text, application

Description automatically generated

After copying, make sure you alter the search paths for libraries in your Delphi-project file, to be relative to the current project folder. The libraries should already be there, otherwise clone them from GitLab. It is also a good moment to clean-up your sources and delete obsolete files. Your previous copy is still there in case you are too rigorous.

Normally the library path would look like this:

*“..\..\..\lib\forms\delphi\_10;..\..\..\lib\excmagic”*

After cleaning-up you have to add the files to your repository.

Graphical user interface, application, Word

Description automatically generated

First, Add all the listed files to your by selecting ‘Stage All’:

Graphical user interface, text, application, email

Description automatically generated

Then you have to commit these files to the repository:

Graphical user interface, text, application, email

Description automatically generated

By checking ‘Push changes directly’ the files will be pushed to your remote repository in GitLab too.

Your local and remote repository is now setup properly.

**Working with SourceTree after changes in your project:**

1. Perform a commit to your local repository first.

Do not check ‘Push changes immediately to origin/main’.



1. Perform a Pull: This ensures that changes by others are merged into your local repository.

A picture containing graphical user interface

Description automatically generated

1. Perform a Push: This will update the remote GitLab repository with your changes.

A picture containing graphical user interface

Description automatically generated

After finishing a Project change you can use the Tag function to add a version number to your current Repository.

This can be used to revert to a specific project version, when debugging

A picture containing graphical user interface

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated