QUILLSOFT NLP VIEWER

Setting development environment.

In this tutorial, we use VS Code as the development environment for both the back end and the front-end projects. However, you can use the editor of your preference.

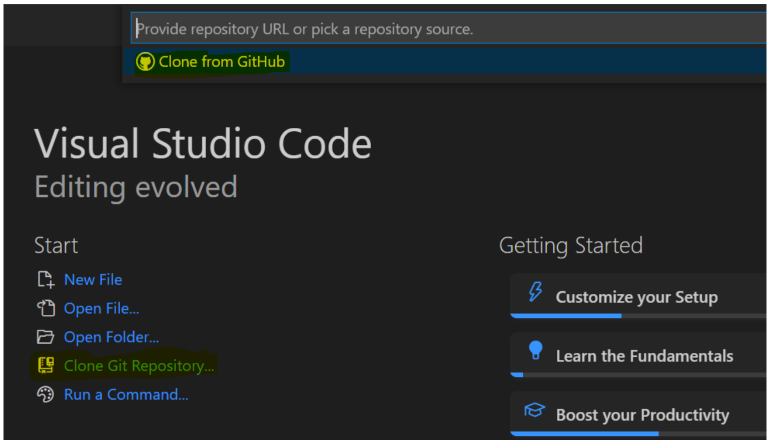
Please, refer to the Readme files of both projects to get a more complete and detailed understanding of the requirements.

Software Requirements

- VS Code (https://code.visualstudio.com/download)
- Python 3.8.5 (https://www.python.org/downloads/)
- Git (https://git-scm.com/downloads)
- JRE
    - Windows (https://www.oracle.com/ca-en/java/technologies/javase-jre8-downloads.html)
    - Linux or Mac (Version 11 to 14)
- WKHTMLTOPDF (https://wkhtmltopdf.org/downloads.html)

GitHub Repositories

The repositories of the source code are hosted on GitHub. You can download the projects manually, using Git Bash, Git GUI, or the Build-In Source Control in Visual Studio Code:



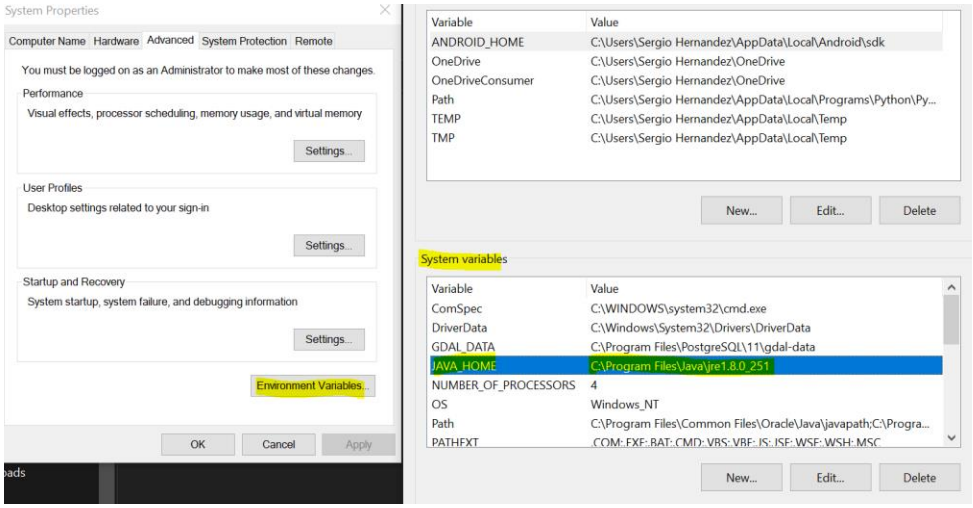
- https://github.com/pherriton/quillsoft-nlp-engine:
- https://github.com/pherriton/quillsoft-nlp-jviewer:

GROBID Service

The GROBID service is a tool the engine uses to extract text and headings from a given PDF file. This service needs to be up as long the as UI is up.

Windows: (https://github.com/kermitt2/grobid/archive/0.6.0.zip)

1. Download and install JRE 8 (https://www.oracle.com/ca-en/java/technologies/javase-jre8-downloads.html)
2. Locate where it downloaded and copy the location: C:\Program Files\Java\jre1.8.0_251.
3. Open Environment Variables and add a system variable path of JAVA_HOME to the location copied.



You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your sign-in

System startup, system failure, and debugging information

Variable	Value
ANDROID_HOME	C:\Users\Sergio Hernandez\AppData\Local\Android\sdk
OneDrive	C:\Users\Sergio Hernandez\OneDrive
OneDriveConsumer	C:\Users\Sergio Hernandez\OneDrive
Path	C:\Users\Sergio Hernandez\AppData\Local\Programs\Python\Py...
TEMP	C:\Users\Sergio Hernandez\AppData\Local\Temp
TMP	C:\Users\Sergio Hernandez\AppData\Local\Temp

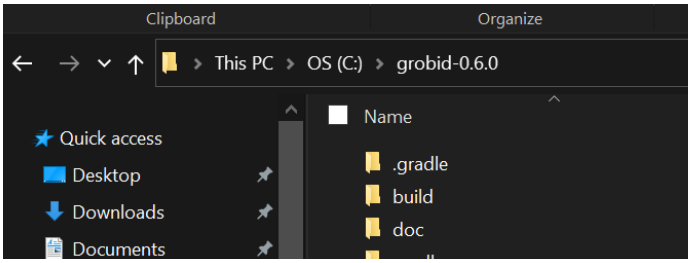
System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
GDAL_DATA	C:\Program Files\PostgreSQL\11\gdal-data
JAVA_HOME	C:\Program Files\Java\jre1.8.0_251
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files\Common Files\Oracle\Java\javapath;C:\Progra...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

Mac/Linux: (https://github.com/kermitt2/grobid/archive/0.6.2.zip)

It requires a JRE version from 11 to 14. There is no need for additional configurations.

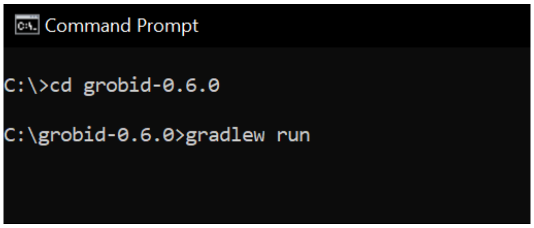
1. Download and extract GROBID on the path of your preference.



This PC > OS (C:) > grobid-0.6.0

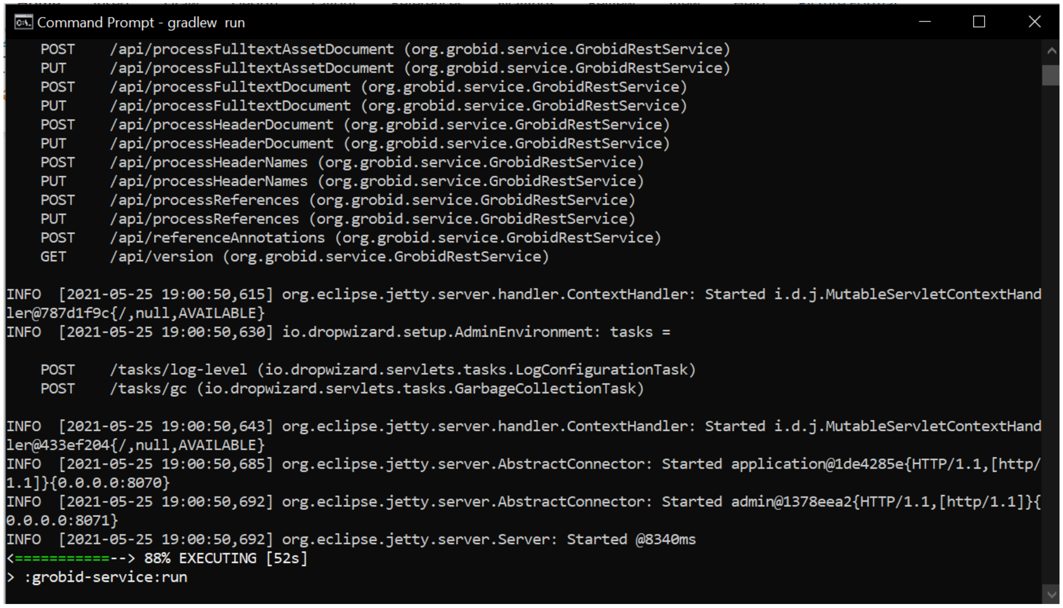
.gradle
build
doc

2. Open a command-line on the location where you have extracted the GROBID service.
3. Run the command "gradlew run"



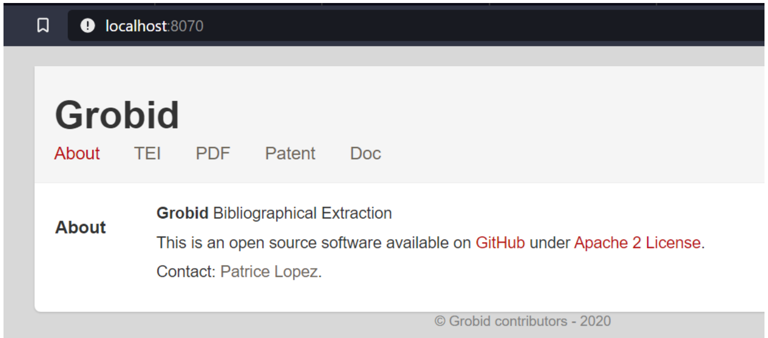
```
C:\>cd grobid-0.6.0

C:\grobid-0.6.0>gradlew run
```

4. If the service is successfully started, you will see a screen like this:



```
    POST    /api/processFulltextAssetDocument (org.grobid.service.GrobidRestService)
    PUT     /api/processFulltextAssetDocument (org.grobid.service.GrobidRestService)
    POST    /api/processFulltextDocument (org.grobid.service.GrobidRestService)
    PUT     /api/processFulltextDocument (org.grobid.service.GrobidRestService)
    POST    /api/processHeaderDocument (org.grobid.service.GrobidRestService)
    PUT     /api/processHeaderDocument (org.grobid.service.GrobidRestService)
    POST    /api/processHeaderNames (org.grobid.service.GrobidRestService)
    PUT     /api/processHeaderNames (org.grobid.service.GrobidRestService)
    POST    /api/processReferences (org.grobid.service.GrobidRestService)
    PUT     /api/processReferences (org.grobid.service.GrobidRestService)
    POST    /api/referenceAnnotations (org.grobid.service.GrobidRestService)
    GET     /api/version (org.grobid.service.GrobidRestService)

INFO  [2021-05-25 19:00:50,615] org.eclipse.jetty.server.handler.ContextHandler: Started i.d.j.MutableServletContextHand
ler@787d1f9c{/,null,AVAILABLE}
INFO  [2021-05-25 19:00:50,630] io.dropwizard.setup.AdminEnvironment: tasks =

    POST    /tasks/log-level (io.dropwizard.servlets.tasks.LogConfigurationTask)
    POST    /tasks/gc (io.dropwizard.servlets.tasks.GarbageCollectionTask)

INFO  [2021-05-25 19:00:50,643] org.eclipse.jetty.server.handler.ContextHandler: Started i.d.j.MutableServletContextHand
ler@433ef204{/,null,AVAILABLE}
INFO  [2021-05-25 19:00:50,685] org.eclipse.jetty.server.AbstractConnector: Started application@1de4285e{HTTP/1.1,[http/
1.1]}{0.0.0.0:8070}
INFO  [2021-05-25 19:00:50,692] org.eclipse.jetty.server.AbstractConnector: Started admin@1378eea2{HTTP/1.1,[http/1.1]}{
0.0.0.0:8071}
INFO  [2021-05-25 19:00:50,692] org.eclipse.jetty.server.Server: Started @8340ms
<===========--> 88% EXECUTING [52s]
> :grobid-service:run
```

5. By default, the service is hosted on port 8070, so you can go to http://localhost:8070 to double-check the service is working properly.



About

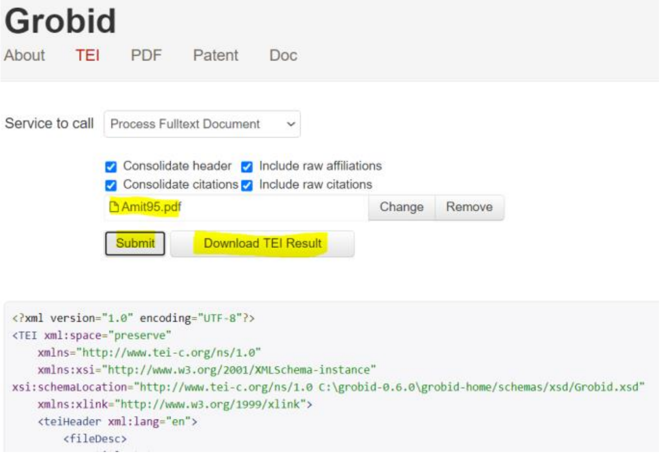
Grobid Bibliographical Extraction

This is an open source software available on GitHub under Apache 2 License.

Contact: Patrice Lopez.

© Grobid contributors - 2020

6. On the TEI tab, you will be able to test the extraction from a PDF file, visualize, and download the generated TEI/XML file.



Service to call: Process Fulltext Document

☑ Consolidate header ☑ Include raw affiliations
☑ Consolidate citations ☑ Include raw citations

Amit95.pdf

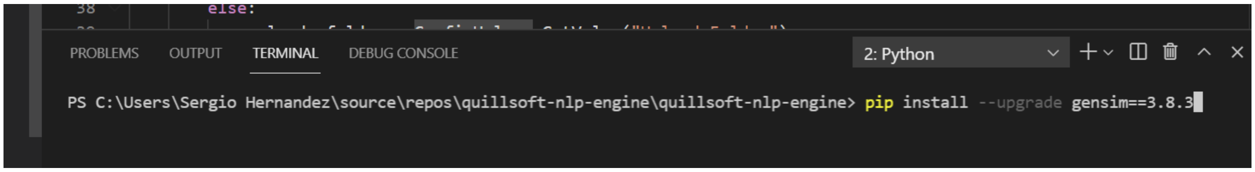
```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xml:space="preserve"
    xmlns="http://www.tei-c.org/ns/1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.tei-c.org/ns/1.0 C:\grobid-0.6.0\grobid-home/schemas/xsd/Grobid.xsd"
    xmlns:xlink="http://www.w3.org/1999/xlink">
    <teiHeader xml:lang="en">
        <fileDesc>
```

Python libraries

Once you have set the development environment for Python and opened the solution, you will find a "requirements.txt" file on the root of the project. Most of the libraries listed in this file can be installed using the command pip install -r requirements.txt.

For those libraries in which the version is detailed in the description of the README file, you may need to perform the installation manually. For example, for new installations, the requirements.txt file is going to download version 4.0.0 of the Gensim library. The current project requires version 3.8.3.

Use the command "pip install --upgrade <<library name>>==<<version>>" in the terminal of the Code Editor.



```
else:
```

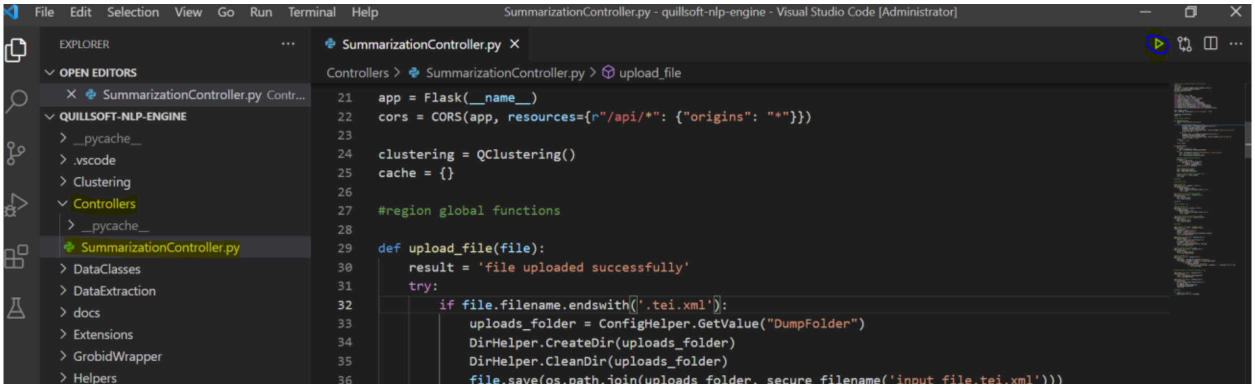
```
PS C:\Users\Sergio Hernandez\source\repos\quillsoft-nlp-engine\quillsoft-nlp-engine> pip install --upgrade gensim==3.8.3
```

Library	**Version**
PyMuPDF	1.18.8
spacy	3.0.3
textacy	0.10.0
gensim	3.8.3

Python artifacts

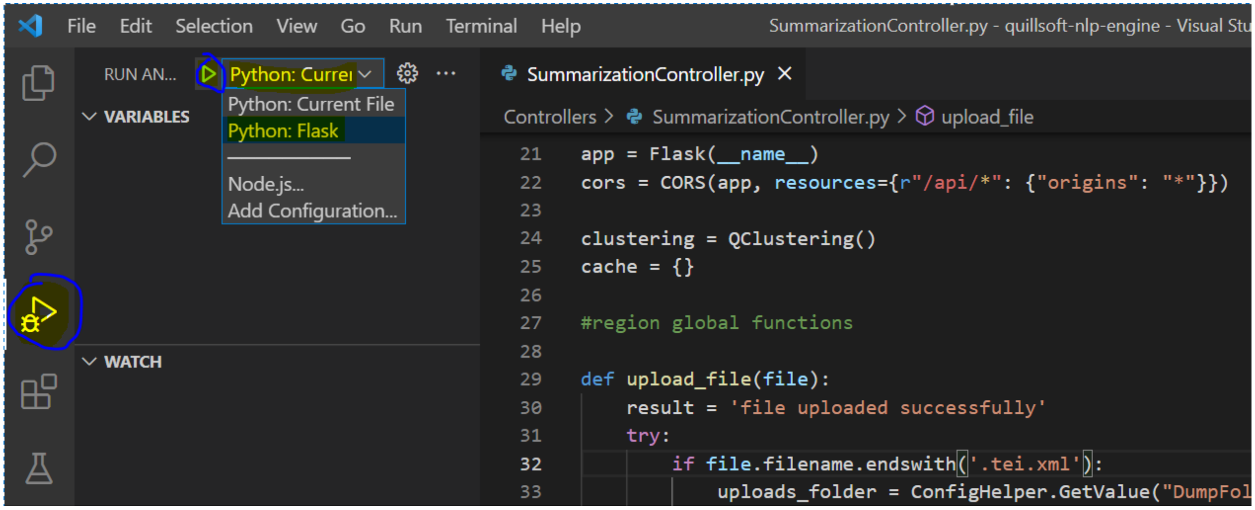
“quillsoft-nlp-engine” contains the backend of the application. There are two deliverables at this point. Either way, you are free to run and compile individual components for which it will be required to go through the Report of the Phase one to get familiar with the source code distribution:

- SummarizationContoller: It is a Flask Service that is required to be up for the UI to be able to communicate with the backend. There are two ways to start the service:
  1. Go to the folder “Controllers” and open the file “SummarizationContoller.py” and then click on the button “Run Python File in Terminal”


File Edit Selection View Go Run Terminal Help
SummarizationController.py - quillsoft-nlp-engine - Visual Studio Code [Administrator]
EXPLORER
OPEN EDITORS
SummarizationController.py Contr...
QUILLSOFT-NLP-ENGINE
pycache
.vscode
Clustering
Controllers
pycache
SummarizationController.py
DataClasses
DataExtraction
docs
Extensions
GrobidWrapper
Helpers
SummarizationController.py
Controllers > SummarizationController.py > upload_file

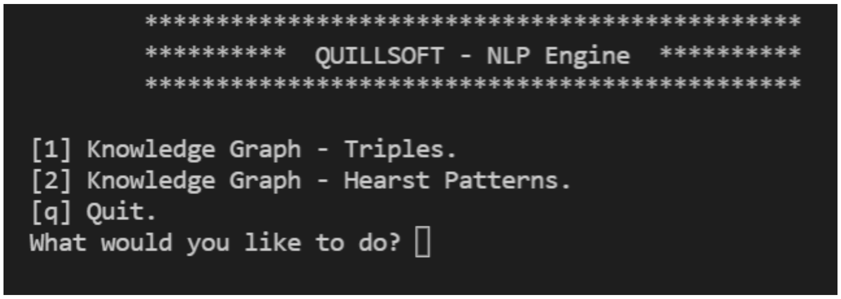
```python
app = Flask(__name__)
cors = CORS(app, resources={r"/api/*": {"origins": "*"}})

clustering = QClustering()
cache = {}

#region global functions

def upload_file(file):
    result = 'file uploaded successfully'
    try:
        if file.filename.endswith('.tei.xml'):
            uploads_folder = ConfigHelper.GetValue("DumpFolder")
            DirHelper.CreateDir(uploads_folder)
            DirHelper.CleanDir(uploads_folder)
            file.save(os.path.join(uploads_folder, secure_filename('input_file.tei.xml')))
```

2. Go to the “Run and Debug Tab”, change the configuration to “Python: Flask” and click on Start Debugging.


File Edit Selection View Go Run Terminal Help
SummarizationController.py - quillsoft-nlp-engine - Visual Stu
RUN AN...
Python: Currei
Python: Current File
Python: Flask
Node.js...
Add Configuration...
VARIABLES
WATCH
SummarizationController.py
Controllers > SummarizationController.py > upload_file

```python
app = Flask(__name__)
cors = CORS(app, resources={r"/api/*": {"origins": "*"}})

clustering = QClustering()
cache = {}

#region global functions

def upload_file(file):
    result = 'file uploaded successfully'
    try:
        if file.filename.endswith('.tei.xml'):
            uploads_folder = ConfigHelper.GetValue("DumpFol
```

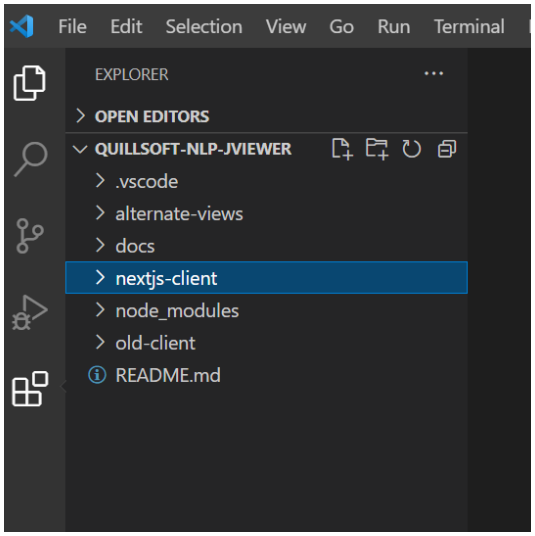
- Init.py: It has a simple console application to allow the testing of Knowledge Graphs generation. This code is not integrated to the main pipeline that the UI calls.
  To run it, simply open the file init.py which is located on the root of the solution, and click on "Run Python File in Terminal" (F5).



```
     ********************************************
     **********  QUILLSOFT - NLP Engine  **********
     ********************************************

[1] Knowledge Graph - Triples.
[2] Knowledge Graph - Hearst Patterns.
[q] Quit.
What would you like to do? 
```

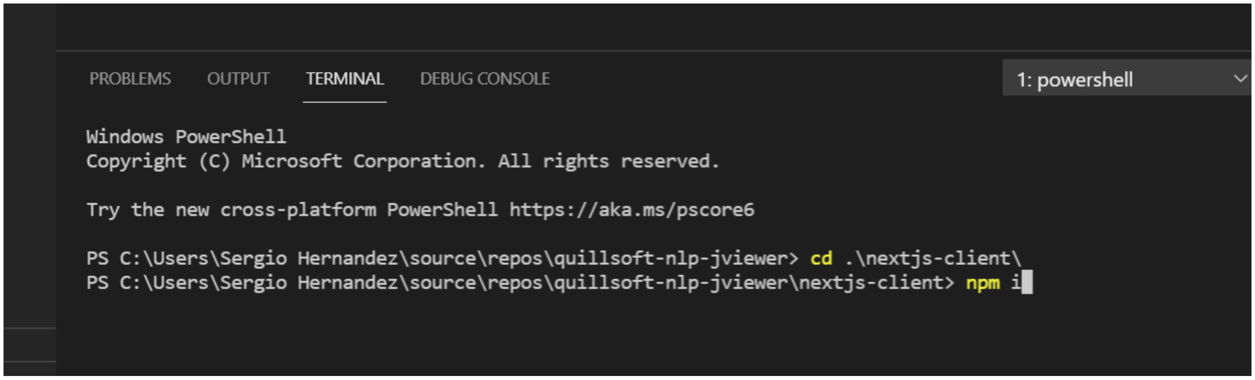
NextJS Application

Initially, you will see four folders on this solution; the final UI is located in the "next-client" directory, but we are going to go briefly through the other ones.



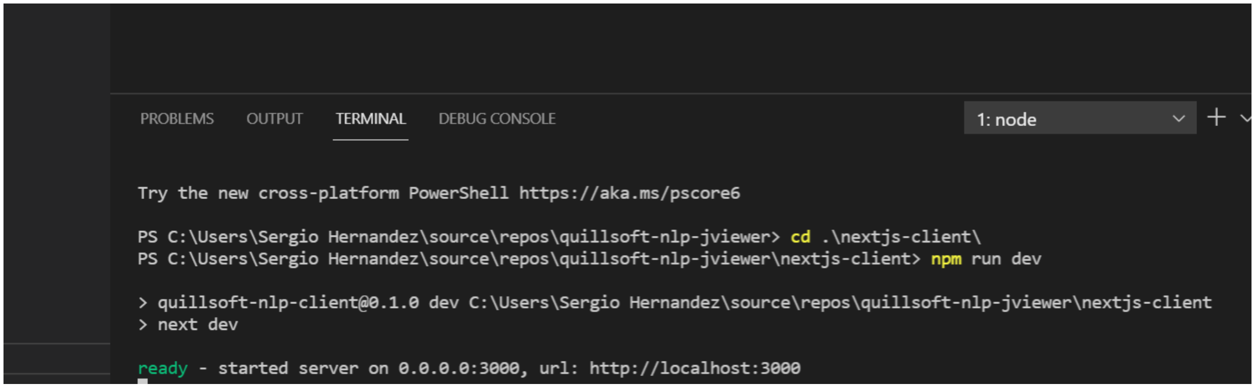
Nextjs-client

1. Before starting the application, be sure the GROBID service and the Flask service are up and running.
2. Run the command “cd nextjs-client” on the Terminal Console.
3. Run the command “npm i” (Only after a new version has been pulled).



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Sergio Hernandez\source\repos\quillsoft-nlp-jviewer> cd .\nextjs-client\
PS C:\Users\Sergio Hernandez\source\repos\quillsoft-nlp-jviewer\nextjs-client> npm i
```

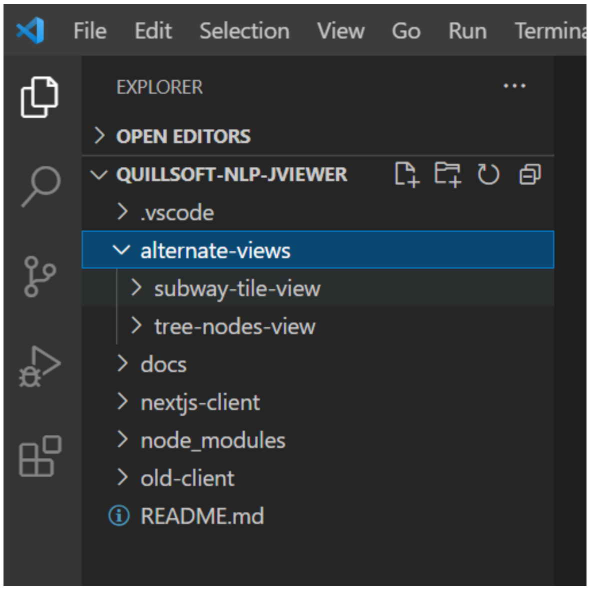
4. Run the command “npm run dev”



```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Sergio Hernandez\source\repos\quillsoft-nlp-jviewer> cd .\nextjs-client\
PS C:\Users\Sergio Hernandez\source\repos\quillsoft-nlp-jviewer\nextjs-client> npm run dev

> quillsoft-nlp-client@0.1.0 dev C:\Users\Sergio Hernandez\source\repos\quillsoft-nlp-jviewer\nextjs-client
> next dev

ready - started server on 0.0.0.0:3000, url: http://localhost:3000
```

5. Open in a browser the URL http://localhost:3000

Alternate-views: It has the source code of a couple of additional views that the development team worked on, but in the end, those views were not implemented. You can refer to the source code for further developments.



```
QUILLSOFT-NLP-JVIEWER
  .vscode
  alternate-views
    subway-tile-view
    tree-nodes-view
  docs
  nextjs-client
  node_modules
  old-client
  README.md
```

To visualize the preview of these views, you can access respectively to the following links:

- Tree Node View: https://kx4o0.csb.app.
- Tile View: https://l06q0.csb.app .

Old Client: It contains the HTML client developed for phase one of the project. It'll run partially since the latest changes of the backend were not updated here.

Docs: It contains this tutorial document and the Results Report with all the details of the second phase of the project.