**Installation instructions**

**Please follow the below instructions to be able to run this tool.**

1. Download/clone the tool from the github repo.
2. Open a terminal and cd into that directory.
3. If needed, create a conda environment with python 3.7
   1. conda create -n pdftool python=3.7
   2. conda activate pdftool
4. Install all of the requirements
   1. pip install –user -r requirements.txt
5. Run the app!
   1. streamlit run tool.py

**General Overview**

This tool is written in python 3.7.5

This tool uses streamlit as it’s core visualization library. It reads PDFs with pdftotext and deals with it’s data management using pandas.

There are three main files to worry about, **app.py, analytics.py, utils.py**.

**Utils**

Utils.py includes multiple useful functions that are reused in terms of dealing with PDF files and how to manage the corresponding data. Each function has a doctoring attached to it that provides a brief description of the function along with the inputs and outputs.

In general, these functions provide assistance in reading pdf files and cleaning the corresponding text taken from it.

|  |  |
| --- | --- |
| Function | Description |
| Clean\_pdf\_page | Takes a list representation of a pdf page and returns the cleaned sentences. |
| read\_pdf\_file | Converts a file to a pdftotext object |
| get\_sections | Get the different sections in a given page |
| calculate\_distance | Calculate the cosine distance between all vectors in two dataframes |
| get\_similar\_sentences | Using scikit-learn's count vectorizer, vectorize the two sets of text and find the best closest ones. |
| clean\_text | Clean a text by lowering the text, removing symbols and stopwords. |

**Analytics**

analytics.py includes functions that pertain to the actual analysis done in the main tool. Functions such as querying the data, finding headers and comparing two dataframes. This file uses utils.py as a base to run its analyses. This file is called by tool.py to generate the output. Each function takes in a list of pages as input and returns a corresponding dataframe visual.

|  |  |
| --- | --- |
| Function | Description |
| get\_words\_in\_sentances | Get all words in the words parameter |
| get\_headers | Find Section headers and sub headers in a dataframe |
| get\_frequent\_words | Find the most common words in a list of pages |
| get\_comparison\_similar\_words | Finds similar sentences between two dataframes |

**Tool**

tool.py is the main backbone of the app, utilizing streamlit to create widgets to be able to perform the functions needed. The app first takes ina pdf file as input and then there is a dropdown that allows the use to select which option they want to display, along with a comparison option at the bottom.

|  |  |
| --- | --- |
| Mode | Description |
| Should, Shall, Must | Scrape through the pdf document and find where the words: Should, Shall and Must are used. Visualize the pages and sections they appear in |
| Headers | Scrape through the PDF document and find all headers and display them |
| Query | Using a custom string, scrape through the pdf document and find where the inputted words are used. Visualize the pages and sections they appear in |
| Section Word Frequencies | At a section level, find which words are most often. |
| PDF Comparison | After taking another PDF input, compare both pdfs according to the similar Should, Shall and Must words used |