# Conceptual Design

UPDATED Wrike Gantt Chart: <https://www.wrike.com/open.htm?id=963685331>

## Functional Decomposition

There are several processes which will need to be accomplished both with and without user input.

1. We will need to capture the users desired inputs and store them, this includes the PDF, epub, word, or text documents of their choosing. If time allows, we would also like to collect the preferred speed and voice of the user and allow the users to input.
2. The document must then be read and stored
3. This file can then be output to the users for editing (Optional)
4. From the edited text file, the system must then take the text file and convert it to speech and output it in the form of an MP3 file (allow Daisy files output if we have the time) to be downloaded for the user

## Case Diagram

* 1. Website, CSS, and html, dropdown to select which input types and a button to submit, this would meet the specification for a website and allows other specifications to be met. The drop-down menus would allow the user to select the appropriate voice and speed, however would may be less accessible to some users
  2. Similar to naturalreaders follow along by highlighting text, this is very accessible and easy to use. This accessibility would most likely come at the cost of some features.
  3. Like apple, long scrolling page – static background clear and easy to follow, scrolling may reduce accessibility of users on smaller devices. Room for many of the target specification features.

1. 1. (PHP)[spatie/pdf-to-text: Extract text from a pdf (github.com)](https://github.com/spatie/pdf-to-text) Uses unfamiliar language and speed is less than required in the target specifications, however within marginal specifications.
   2. (Python) [Convert PDF to TXT file using Python - AskPython](https://www.askpython.com/python/examples/convert-pdf-to-txt#:~:text=Steps%20to%20Convert%20PDF%20to%20TXT%20in%20Python,IDLE%20and%20press%20keys%20ctrl%20%2B%20N.%20) familiar language would allow the code to be adjusted to attempt to meet the desired target specification.
   3. (JavaScript) [How to convert PDF to Text (extract text from PDF) with JavaScript | Our Code World](https://ourcodeworld.com/articles/read/405/how-to-convert-pdf-to-text-extract-text-from-pdf-with-javascript#:~:text=How%20to%20convert%20PDF%20to%20Text%20%28extract%20text,4%204.%20Extracting%20text%20from%20multiple%20pages%20) Unfamiliar language however is built into web code will reduce complexity of hooks and allow for fastest form of program.

Pierre:

1. (xVASynth) Text to speech program: [xVASynth 2 - SKVA Synth at Skyrim Special Edition Nexus - Mods and Community (nexusmods.com)](https://www.nexusmods.com/skyrimspecialedition/mods/44184) - This allows the synthesis of any voice when provided with enough sample size. Would allow for maximum choice of voice, but heightened complexity.
2. (Access) Create a database to draw from and organize chapters [Microsoft Access Application Software | Create Database Apps](https://www.microsoft.com/en-ca/microsoft-365/access) - This would increase complexity, however overall organisation and presentation to end users would be simpler.
3. (OpenSource) Open source pdf to text applications [PDF2Text Pilot download | SourceForge.net](https://sourceforge.net/projects/pdf2textpilot/) - This would forgo the requirement of having a web based application, and rely on executables instead. This would not have the same level of convenience; however it would decrease development time, and could be included as a download from a website.

Wanis Hassan:

* 1. Convert pdf to text with Python: <https://www.pdftron.com/ppc/python-pdf-library/?utm_source=google&utm_medium=cpc&utm_campaign=PDFTron_Google_Search_NB_SDK_Language_Framework&utm_content=python-library&utm_term=python%20pdf%20library&gclid=CjwKCAjw7eSZBhB8EiwA60kCWwxtIGXMiYYeW-dd2Q9dv8oLALtFZZfq0K0M3jgIvSPRKSgkldVCqRoCgvoQAvD_BwE> – This allows us to convert any pdf document to a text file with the use of a programming language our team is comfortable with.
  2. Convert text to speech in Python: <https://pypi.org/project/gTTS/> - This allows us to convert the text that we had previously converted from pdf to mp3 with the same language we used for the last step, so maintaining both codes will be consistent.
  3. Open-source image to pdf converter: <https://dawnlabs.github.io/alchemy/> - This could be used if we end up using executables, it will allow us to convert images to pdf.

Haonan:

a. Convert image to pdf: <https://datatofish.com/images-to-pdf-python/> - It helps our group to convert images first before converting pdf to speech.

b. (Language Translator): <https://techvidvan.com/tutorials/python-language-translator/> - It can make our project available in multiple languages.

c. Convert text to speech: <https://www.codingem.com/python-text-to-speech/> - This web can easily help us to build our mind about how to convert text to speech.

1. Will be implemented if we have enough time
2. 1. (Node.js)[eheikes/tts: Tools to convert text to speech (github.com)](https://github.com/eheikes/tts) JavaScript library will be harder to work with but is web code that is more efficient and allows for better integration.
   2. (java) [Create voice audio files  |  Cloud Text-to-Speech Documentation  |  Google Cloud](https://cloud.google.com/text-to-speech/docs/create-audio#text-to-speech-text-java)

Google API will be of high level and allows to choose between multiple voices will need to be embedded in web code of be its our .exe executable. Allows to configure the output of speech in a variety of ways: selecting a unique voice or modulating the output in pitch, volume, speaking rate, and sample rate.

* 1. (Python) [Create voice audio files  |  Cloud Text-to-Speech Documentation  |  Google Cloud](https://cloud.google.com/text-to-speech/docs/create-audio#text-to-speech-text-python) Google API will be of high level and allows to choose between multiple voices, Python is the language most of the group members have experience with. Allows to configure the output of speech in a variety of ways: selecting a unique voice or modulating the output in pitch, volume, speaking rate, and sample rate.

## Analyze and evaluate

The team has decided that given the members experience with python It would be in our best interest to build a fully python-based executable. This would not meet the target specification of being web-based application, however, in time we may be able to migrate the python code to an html website. Given that none of the team has any JavaScript experience and that the software is intended to primarily be used by librarian, who have work laptops that can download the executable, this is the best option given the limited work time the team has available for the project. This decision means that the most relevant ideas will be those which meet target specifications and make use of python libraries. This also means that we would need to develop a python GUI most likely built using TKinter.

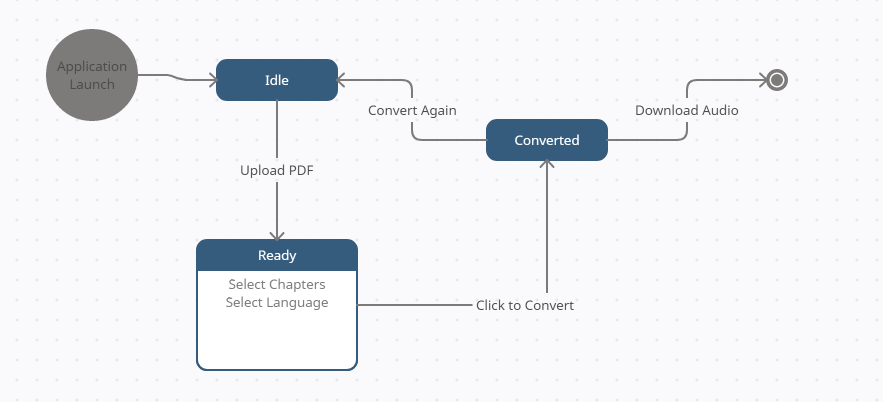
## Promising solutions

1. Convert text to speech in Python: https://pypi.org/project/gTTS/ - This allows us to convert the text that we had previously converted from pdf to mp3 with the same language we used for the last step, so maintaining both codes will be consistent.
2. (Python) Convert PDF to TXT file using Python - AskPython familiar language would allow the code to be adjusted to attempt to meet the desired target specification.

## Group design

The current design will be fully built on python. The user will download the executable on their device and interact with a GUI built using TKinter. Once the PDF file is input into the program would transform the file into a TXT file using a python script using an external module named PyPDF2 with the tutorial for its implementation found on AskPython. Next using the GUI the user will hopefully be able to edit the TXT document and resubmit the file. This file is converted into an MP3 using the pypi.org project library and is downloaded directly into the user’s designated folder.

## Visual Representation

State Diagram:  


Rather than implementing a web-based solution, the application will be fully self contained. The benefits of this are as follows:

This will allow for offline access to the application.

This will allow students to be given the program and use it independently.

This will simplify the design process and have no dependencies on external downtimes.

The downsides of this are as follows:

The program will not be easily accessible on new computers (This can be worked around by uploading the program to a download site and linking it.)

The program will be limited to python libraries. This makes future upgrades and expansions limited to existing python libraries.