- [x] ~~Make list of stuff that needs to be redacted~~
- [x] ~~Find cover letters and resumes to test on~~
    - [ ] Different file formats? Ex. jpg, pdf, doc, doocx (or limit)
    - [ ] Resume examples: https://www.resume.com/sample/engineering-resumes-sample
    - [ ] Cover Letter examples: https://career.virginia.edu/resumes/writing-cover-letter/cover-letter-sample
    - [ ]
- [ ] Bryce - add code in CoreNLP algo to redact pronouns
    - [ ] Regex
- [ ] Anjala - get python to read a pdf
    - [ ] Make it work with both nlp programs
    - [ ] Remove weird characters
    - [ ] Try to format back into pdf - later
    - [x] ~~Regex stuff~~
- [ ] Matt - Look into custom tagging on spacy - https://spacy.io/
    - [ ] Spacy issues: names, number in addresses
    - [ ] Try Regex
- [ ] Ask neha about customizing tagging system?
- [ ] Get a list of names and see how many each can redact
    - [ ] https://github.com/smashew/NameDatabases/blob/master/NamesDatabases/first%20names/all.txt
- [ ]

Meeting Notes:

- Spacy issues: names, number in addresses
- CoreNLP issues:
- Combining both, or CoreNLP seems like best option
- Test on complicated names/figure out solution
    - Remove names, addresses, other personal info input into
    - Have user confirm redaction
- Customize a tagging system

Regex

- [x] ~~Email - remove anything with an @ symbol~~
- [ ] Phone Numbers
- [ ] Location Info - don't think there is a regex pattern to find locations, probably need a list of states, countries, state abbreviations to check against
- [ ] Address Validation with re - https://stackoverflow.com/questions/11456670/regular-expression-for-address-field-validation#:~:text=Here%20is%20the,in%20your%20data.

- [ ] Dates with dashes - https://regexland.com/regex-dates/
    - This is for YYYY-MM-DD or YYYYMMDD
    - ^\d{2}\-\d{2}\-\d{4} for DD-MM-YYYY

Things to Redact

- **Names**
- **Emails**
- **Pronouns**
- **Location info - cities, states, addresses**
- Dates - graduation dates
- **Phone numbers**
- Gender (i.e. “I am a [Gender]”, Replace man/woman/etc.)
- Maybe:
    - School names
      Citizenship status

(Check when accuracy of redaction is good)
If accuracy not good, write approx percent accuracy next to item

Spacy	CoreNLP
Dates (slashes and text, but not dashes) Pronouns - somewhat manually Email addresses - manually	Names Addresses Pronouns - manually works Location info

Stuff to make work - location info

CoreNLP

- [x] ~~Names~~
- [ ] Emails
- [ ] Pronouns
- [x] ~~Location Info~~
- [x] ~~Dates~~
- [ ] Phone Numbers
- [ ] Gender

Spacy

- [ ] Names
- [x] ~~Emails~~
- [ ] Pronouns
- [ ] Location Info

- [ ] Dates
- [ ] Phone Numbers
- [ ] Gender

Spacy Custom Tokenization

- Possible solution 1: Set entity annotations by creating a new entity using the tokens generated by nlp after running text through spacy
    - entity_name = Span(doc, start_loc, end_loc, label="LABEL")
- Spacy's tokenizer uses English grammar rules
- Possible solution 2: Adding special case tokenization rules
    - Before: print([w.text for w in doc]) # ['gimme', 'that']
    - special_case = [{ORTH: "gim"}, {ORTH: "me"}]
      nlp.tokenizer.add_special_case("gimme", special_case)
    - After: print([w.text for w in nlp("gimme that")]) # ['gim', 'me', 'that']
- Possible solution 3: Modify existing rules sets using regex
    - suffixes = nlp.Defaults.suffixes + [r'''-+$''',]
- Possible solution 4: Spacy matcher - https://spacy.io/api/matcher
    - Similar to solution 2
    - Could identify emails, numbers, etc.