Design Decisions and Discussion

General Alexa design

- Alexa development kit makes it difficult for multiple people to edit files at the same time, outside integration with IDEs is spotty and testing is difficult, so development is slower and requires greater coordination between developers, very restricted git integration
- Numbered menus not possible in a logical and expected way
    - Single commands like "one" and "two" triggered skill-wide intents, instead of being context-based
    - Got around this with specific context commands like "section {number}"
- Use of S3 due to its ease of integration with Amazon/Alexa
- Effectively using data persistence for UX across sessions, remembering information like user name, policy reader location, etc.
- Alexa device has a hard time correctly parsing complex user input, such as a filename or long string of words with any symbols (e.g. /, &, #, etc)
    - Expecting user to remember and repeat a lot of text, such as section titles
- Alexa having a process for automatically prompting to fill slot types
    - If a piece of information is marked as required, Alexa will continuously prompt the user until the slot is filled. Does not allow for conditional slot requirements

Policy Reader design

- For long blocks of text, performs better with an audio player as opposed to TTS
    - Buffer times, configuring TTS to sound natural, can browse audio sections in a more expected way (rewind, FFW, skip around)
- Best way to parse a policy?
    - We used XML because it's easy to parse, format, and can be useful for both long and short policies + extensible
- Having to make the choice of how often to query user to "continue"
    - No way to just automatically continue reading through all sections, Alexa requires a continuous user prompting in order to do any speech output — tradeoff between reading longer amounts with no prompting, or finer granularity but more prompting
    - If we had it read through one large block of text, would run into buffer time issues, lose ability to track location in document
- Reading options like speeding up and slowing down were already built into the Alexa
    - Considered options like other voices, SSML tags, but the default Alexa voice works pretty well for being understandable and interesting to listen to — with more time, probably would be interesting to implement more user choice on playback styles
- Went with a more global set of commands instead of context-based, all commands can be issued at any time instead of being dependent on Alexa's current task
- Allowing for piecemeal acceptance of policy instead of all or nothing

- Made a custom privacy policy class for ease of development, avoids a lot of programming repetition and makes for more readable code
    - More 'one-liners', less dense code as opposed to lots of dictionary accesses

Privacy Manager design

- Alexa cannot actually access raw user audio, record from users, only receives a JSON output of what Amazon parses from the user speech
- Easier to use S3 with AWS, but in the future a more traditional relational database would be more appropriate, especially with cascading deletes
- When requesting file access from another user, access is requested for all files by default and then the requestee can choose specific files
    - Considered idea of a file browser for secondary users but it runs into privacy issues very quickly, also difficult to set up and likely confusing
- Parsing filenames is difficult, same issue with not being able to use a numbered prompt easily, Alexa has a hard time parsing anything that isn't a word
- Deciding best way to store user preferences and settings
    - Went with a JSON dictionary instead of INI file, useful with S3 persistence, JSON allows more dynamic settings
- Used a basic login system, for demonstration purposes just used built-in Amazon names slot types, Alexa parses those more easily than if we went with a more freeform username or number system

- Accept and deny had to be separate intents, because both commands require different amounts of information and Alexa doesn't allow for conditional slot requirements
    - Deny just requires a name, whereas accept requires both a name AND file names
- Audio playback cannot be tested within the development environment/web browser, audio can only be played on an actual Alexa device or smartphone
- File deletion dialog would be extremely simple to implement