Verification

Instructions

A separate description of the tests you ran on your program to convince yourself that it is indeed correct. Also describe any cases for which your program is known not to work correctly.

Thoroughness

Development and testing was only done using the `simple` config files, which are made up of 3 SuperPeers each with 2 Leaf Peers.

- Registration: tested with a Peer's owned files. Largely no code changes from PA1 and PA2.
- Deregistration: tested with a Peer's owned files. Files are **not** deleted when they are within the `owned/` directory, letting us deregister and register as we please.
- Query: requested files that a Leaf Peer under a **different** SuperPeer to test the forwarding (hops). Had multiple Peers (>3) request the same file. Peers within the same cluster (i.e., under the same SuperPeer) would typically be used to download the file.
- Consistency:
  - In the PULL consistency model, I had the multiple Peers above query the same file, and thus each would periodically poll the origin server. I then had the origin server deregister the file, resulting in a "deleted" message being returned on the next poll. Those Peers subsequently deregistered their file and **deleted the file from their `downloads/` directory**.

    This tested the consistency across at least one hop (between SuperPeers) and tested the periodic polling (and reaction to an origin server deleting its file).

    This experiment was repeated with file *modifications* as well, which correctly returned a "outofdate" response. Otherwise, the response was "uptodate".
  - In the PUSH consistency model, testing was easier. I had the origin server both modify and delete a file, which resulted in a "invalidate" broadcast, forcing the Peers to deregister and discard their copies.

How I Tested

I found it easiest to use a terminal multiplexer (e.g., `tmux`) to open several panes, and see how initiating one type of request resulted in several print statements for different Peers and SuperPeers.