Surya Rani

Design doc asg1

Main:

- Receive message from client
- Parse through message to see what is being called either put, get or head
    - Parsing will require to use strtok probably to be able to delimit each header by \r\n
    - Once we get the first header we need to find the function name and the file name
    - Check if each header after that follows the correct http conventions if not throw error
    - Get content length if request has it and store it
- If correctly formatted call that function method
- For each function it will require different arguments
- Check request to make sure it is acceptable:
- 1.Check if function name is either get put or head
- 2. Make sure file has / in front of it
- 3. check if file name is 27 characters or less not including the /
- 4. check if file name is entirely alphanumeric or has "-" or "_"
- 5. check each header and make sure that it is a valid header %s: %s or is Content-Length: %d
-

Put:

- Requires the client socket to be able to receive the data it wants to put into file
- Requires content length, filename
- Want to be able to request to write to a file
- Open a preexisting file, or create one if it doesn't exist
- Read in file size to make a buffer, using the content length that we received from request
- We then must call receive again to get the data from the client socket to be able to place into the buffer
- Since we don't want to fully trust recv because it might not get all the bytes at once we want to set a counter that will add to itself each time we receive in a loop and keep receiving till it hits the file size or the content len
- Write to file with buffer contents
- Return code 201 if everything works out and creates a file of that name with the contents
- Else we check for errors and return the appropriate error code
- Error check files with both stat and ERRNO

Get:

- Requires the client socket to be able to send data to it
- Requires the file name to get data from as well
- Make sure file exists and can be opened using stat or open
- Find the size of the file first to be able to send in the response to the client

- Use filename to see if directory has it and if it does open it and start continually reading data and writing to the client in a loop
- If file is not in the directory return 404
- If file is forbidden to access then return 403
- Any other error is 500
- Return code ok with 200 if everything works and also send file size
- Error check files with both stat and ERRNO

Head:

- Same thing as get but do not send the contents of the file back to the server
- Just send the file size
- If filename bad file 400
- If forbidden 403
- If filename doesn exist 404
- If everything is good then send ok code 200
- Error check files with both stat and ERRNO