Hva som fungerer:

Programmet fungerer og får forventet resultat i fra alle testene.

Connection.c:

Implementeringen av TCP-koden i connection.c er basert på cbra video om TCP og beskrivelsene av funksjonene i connection.h

Funksjonen tcp_connect oppretter en socket ved hjelp av socket-funksjonen og initialiserer en sockaddr_in-struktur med vertsadresse og port. Deretter bruker den inet_pton-funksjonen for å konvertere vertsnavnet til et nettverksbinært format. Til slutt bruker den connect-funksjonen for å opprette en TCP-tilkobling til den angitte vertsadressen og porten. Den returnerer socketen for den opprettede TCP-tilkoblingen

Funksjonen tcp_read bruker read-funksjonen for å lese data fra socketen og lagrer resultatet i bufferen. Hvis lesingen mislykkes, skrives en feilmelding ut til stderr. Deretter returnerer funksjonen antall bytes som ble lest.

tcp_write bruker write-funksjonen for å skrive data fra bufferen til socketen. Hvis skrivingen mislykkes, skrives en feilmelding ut til stderr. Deretter returnerer funksjonen antall bytes som ble skrevet.

tcp_write_loop bruker en while løkke for å sikre at alle bytes blir skrevet til socketen. Den kaller tcp_write-funksjonen gjentatte ganger med en del av bufferen til alle bytes er skrevet. Hvis skrivingen mislykkes, returneres feilkoden. Ellers returnerer funksjonen totalt antall bytes som ble skrevet.

Funksjonen tcp_close lukker tilkoblingen ved å kalle close-funksjonen.

tcp_create_and_listen oppretter en socket ved hjelp av socket-funksjonen og binder den til INADDR_ANY og den angitte porten ved hjelp av bind-funksjonen. Hvis bindingen mislykkes, skrives en feilmelding ut til stderr. Deretter bruker den listen-funksjonen for å lytte etter innkommende tilkoblinger på socketen.

tcp_accept aksepterer en innkommende tilkobling. Funksjonen bruker accept-funksjonen for å godta en tilkobling og returnerer en ny socket descriptor for den aksepterte tilkoblingen. Hvis aksepteringen mislykkes, skrives en feilmelding ut til stderr.

tcp_wait bruker select-funksjonen for å vente på en aktivitet på socketene. Hvis ventingen mislykkes, skrives en feilmelding ut til stderr. Deretter returnerer funksjonen antall aktive socketer.

tcp_wait_timeout bruker select-funksjonen med en tidsavbruddsverdi for å begrense ventetiden. Hvis ventingen mislykkes, skrives en feilmelding ut til stderr. Deretter returnerer funksjonen antall aktive socketer.

Hendelsesløkken i proxy.c og konverteringen:
Hendelsesløkken venter på aktivitet på hvilken som helst av de tilkoblede klientene eller på serverens socket.

Ved aktivitet på serverens socket, antas det at en ny klient prøver å koble til. Da kalles handleNewClient-funksjonen. Denne funksjonen godtar den nye tilkoblingen, oppdaterer venteSet (et sett med sockets som det venter på aktivitet fra), og initialiserer Client-strukten for den nye klienten. Hvis det allerede er maksimalt antall aktive klienter, vil den nye tilkoblingen bli stengt.

Ved aktivitet på en av klientens socket, antas det at klienten har sendt data. handleClient-funksjonen kalles da for å håndtere den situasjonen. Denne funksjonen leser inn data fra klienten og legger det til klientens buffer. Hvis meldingen er komplett, opprettes en Record-struktur fra meldingen, og forwardMessage-funksjonen kalles for å sende meldingen videre til den tiltenkte klienten.

Meldinger som kommer fra klienten er enten i XML eller binær, avhengig av klientens mldType. I handleClient() blir meldingen konvertert til en Record ved hjelp av enten XMLtoRecord() eller BinaryToRecord(), avhengig av meldingstypen. Når en melding er konvertert til en Record, blir den sendt til forwardMessage(), som finner klienten meldingen skal videresendes til, konverterer Record tilbake til enten XML eller binær, avhengig av destinasjons klientens meldingstype, og sender meldingen til den.

XMLtoRecord funksjonen begynner med å initialisere en ny Record og looper gjennom bufferen tegn for tegn, og ser etter XML-tagger. Når en tag er funnet, leser den navnet og verdien, og setter de tilsvarende feltene i Record basert på tagens navn. Hvis den finner en </record> tag, avslutter den løkken og returnerer Record. Hvis den ikke finner denne taggen innen bufferen er ferdig, slettes Record og funksjonen returnerer NULL.

BinaryToRecord funksjonen fungerer på en lignende måte, men forventer data i et binært format i stedet for XML. Den begynner med å lese en byte som inneholder flagger for hvilke attributter som er inkludert i meldingen. Den leser deretter hver av disse attributtene basert på flaggene, og setter de tilsvarende feltene i Record. Hvis den når slutten av bufferen før alle attributtene er lest, slettes Record og funksjonen returnerer NULL.