CN Project Phase2 report

Team members

- 陳柏諺 B09902061
- 董瑋 B09902063

Implementation details

- TCP connect
  定義一個結構體 package 使 TCP server 和 client 間的傳輸標準化，每次讀寫只需恰一個 package 的大小，且其中的屬性能提供 server 和 client 必要資訊。與 Phase 1 使用同樣的 TCP 連線方式，透過 socket 實現 server 和 client 間的傳輸。

```
struct package{
    int type, buf_size;
    char buf[2048], sender[64], recver[64];
    time_t Time;
};
```

- HTTP response/request
  - accept:
    accept() 到連線時利用 select() 判斷是否有新寫入能讀取，讀取 request 且 response 後 close() 這次的連線再等待下一個連線。若中途 send/recv 等有出現錯誤，在 close() 這次的連線後強迫下次連線時重新登入帳號。
  - header:
    用 read 每次讀取一個 char 並將其加進一個 string 中，當讀到連續的 \r\n\r\n 時代表 http 表頭結束，利用 find() 尋找空格或關鍵字並用 substr() 取出有用的部分，例如：Method, Content-length, boundary 等。
  - GET:
    從表頭得到 GET 的網址後利用 package 的 sender 和 recver 填入檔案路徑並向 TCP server 端請求檔案，TCP server 接到請求後會依照檔案大小(不存在則為 -1)返回一個 package，TCP client 依此大小接收並填入 HTTP response header 的 Content-length 中，在接收到剩餘 package 的同時將其內容直接傳給 HTTP client。
  - POST:
    依 Content-length 大小讀取 HTTP request content，並照不同的 post action 處理。
    - username / add_friend / del_friend：將 content 內的名字利用 package 傳給 TCP server 並接收回覆，再依回覆回應 browser。
    - list_friend：HTTP server 接到後會向 TCP server 詢問好友清單並將其以文字格式置於 homepage.html 的尾端再傳給 browser。
    - homepage / logout：由 HTTP server 端向 TCP server 讀取 homepage.html 或 index.html。
    - send_message：將文字資料包裝到package中傳到server並記錄起來
    - send_image / send_file：先將檔名和檔案大小傳給server，接著再不斷傳送檔案到server端由server紀錄下來。
    - updaate / view_history：傳送一個package要求把聊天紀錄回傳給client，如果沒有指定要看多少則紀錄就會回傳最近30則訊息給client。

- chat history 和資料儲存方式
  對於每個使用者都建立一個專屬資料夾，底下再對所有好友再各自建立一個資料夾，裡面放著他和其他人聊天的內容以及傳輸的檔案。TCP server 在收到 TCP client 傳送的 package 後直接將整個 package 寫入位於 /server_dir/sender/recver/ 以及 /server_dir/recver/sender/ 中的兩個 chat 檔案，收到 image 或 file 時也是同時寫入兩個位置。當需要歷史紀錄時則進入 chat 中將整個 package 讀出並傳送至 TCP client，再由 TCP client 將文字或 url 附在 html 檔的後面傳至 web。傳輸的檔案和圖片會以 Unix 時間作為新檔名加上原本的副檔名存在資料夾中，可以避免檔名相同而被覆蓋的情況，在 chat 中會同時儲存新舊兩個檔名以方便在 web 的聊天紀錄中分辨，但在 console mode 的聊天紀錄中為了與下載後的檔名對照，因此會直接顯示新檔名。
- auto refresh
  在 chatroom.html 中利用 javascript 的 setInterval() 和 submit() 函式使在 chatroom 時每隔 30 秒便會自動發送 post /update。但若在 post /update 時恰好有檔案在傳輸或訊息尚未輸入完成可能會導致檔案上傳失敗以及訊息消失。
- client console mode
  移除 client 中與 HTTP 相關的部分，改為直接由標準輸入讀取、標準輸出回應。
- Server 連線方式
  利用 socket 建立連線，同時搭配 IO mutiplexing 來讓不同 client 都能同時被服務到，且在上傳檔案時也是如此。故不會出現單一 client 上傳大型檔案導致 server 被其占用，而其他人無法使用的狀況出現。

Demo link

https://youtu.be/owIcsIdkh0k

Work division

- 陳柏諺：client端程式撰寫，concole mode程式撰寫，CSS和JS撰寫
- 董瑋：server端程式撰寫，concole mode程式撰寫，HTML撰寫