CAB403: Assignment 1 Documentation

# Authors:

Shaun Karran (n8958815) – Contributed Equally

Lachlan Cesca (n8580413) – Contributed Equally

# Statement of Completeness:

All tasks were attempted and completed.

One issue did arise which couldn’t be fixed was that the server could not exit gracefully if any and only if there were multiple clients on the login screen. Once users were logged in, the server would be able exit gracefully without an issue.

As it wasn’t entirely clear which way the alphabetical ordering for the leader board was to work, if multiple users had the same score. We opted for printing the user with the lowest letter (A) first which would result them being further down on the leader board. This could be easily changed in one line if it was intended the other way.

# Description of important Data Structures:

typedef struct{

char word[MAX\_WORD\_LENGTH];

char adjective[MAX\_WORD\_LENGTH];

}Word;

“Word” Data structure is used to have an array of possible hangman words while keep the word and the adjective word describing the word together, without having two separate arrays.

typedef struct Leaderboard {

Score \*first;

Score \*last;

int num\_scores;

} Leaderboard;

“Leaderboard” which is the header for the linked list data structure that keeps track of all the users scores. The member variables include the first or the head of the linked list, the last (not entirely necessary but is useful to keep track of in some purposes) and the number of scores in the leaderboard (num\_scores).

“Score” is the linked list that keeps track of the users scores and points to the next score in the chain/list. Member variables include the username belonging to the score (char username[]), number of games played (int games\_played), number of games won (int games\_won) and the next score in the list (Score \*next).

typedef struct Score {

char username[USERNAME\_LENGTH];

int games\_played;

int games\_won;

struct Score \*next;

} Score;

typedef struct{

int guess\_count;

int number\_guesses;

char word1[MAX\_WORD\_LENGTH];

char word2[MAX\_WORD\_LENGTH];

int len\_word1;

int len\_word2;

char guessed\_word1\_portion[MAX\_WORD\_LENGTH];

char guessed\_word2\_portion[MAX\_WORD\_LENGTH];

char guesses\_made[26];

} Game;

“Game” struct is used to keep track of the current game of hangman being played. It would keep track of the number of guesses made (guess\_count), the number of guesses the user was allowed to make for the provided word (number\_guesses), the two parts of the word to guess (word1 and word2) and the length of those words (len\_word1 and len\_word2), the string displayed to the user with the unguessed letters blanked out(guessed\_word1\_portion and guessed\_word2\_portion) and the guesses the user has made so far.

typedef struct {

int sock\_fd;

bool connected;

char username[USERNAME\_LENGTH];

char password[PASSWORD\_LENGTH];

} Client\_Info;

“Client\_Info” data structure is used to keep track of the users/clients connected to the server. The member variables include the socket which the client is connected to (int sock\_fd), whether the user is connected still or not (bool connected) and the username and password of the client (char username[] and char password[]).

# Description of Synchronisation Primitives: