Funciones Generales del Sistema:

1. **Insertar registro:**

Cliente:

Se crea una estructura dogType y se le asigna espacio de memoria, se envían los datos insertados por el usuario a través de send, recibe el mensaje de confirmación del servidor, se pide al usuario oprimir cualquier tecla para continuar.

Servidor:

Se crea una estructura dogType y se le asigna espacio en memoria, revive los datos del cliente, guarda la mascota al final del archivo dataDogs.dat, libera la memoria, guarda en el archivo serverDogs.log la información del registro ingresado y envía el mensaje de confirmación, disminuye en uno el número de clientes.

2. **Ver registro:**

Cliente:

Recibe del servidor el número de mascotas, imprime el número de mascotas, se pide seleccionar el número de registro, se envía al servidor el número del registro, se preguntas si se quiere abrir la historia clínica, se envía la decisión seleccionada por el usuario, si la decisión es si se recibe el archivo donde está guardada la historia clínica, abre el archivo de la historia clínica, envía la historia clínica modificada al servidor, recibe el mensaje del servidor y lo imprime, se pide al usuario oprimir cualquier tecla para continuar.

Servidor:

Envía el número de registros al cliente, recibe el número de registro a mostrar, recibe la decisión de abrir la historia clínica, si la decisión es si, lee de la base de datos el registro, concatena los datos de la mascota para nombrar el archivo, crea el archivo, abre el archivo, si el archivo es nulo abre el archivo abre el archivo, imprime los datos de la mascota, cierra el archivo, envía el archivo al cliente, recibe el archivo modificado, guarda en el archivo serverDogs.log la información del registro ingresado y envía el mensaje de confirmación, disminuye en uno el número de clientes.

3. Eliminar registro:

Cliente:

Recibe del servidor el número de mascotas, imprime el número de mascotas, se pide seleccionar el número de registro a eliminar, se envía al servidor el número del registro, recibe el mensaje del servidor y lo imprime, se pide al usuario oprimir cualquier tecla para continuar.

Servidor:

Envía el número de registros al cliente, recibe el número de registro a eliminar, elimina la mascota del registro, desconectado la mascota y eliminándola de la base de datos, disminuye uno el número de registros guarda en el archivo serverDogs.log la información del registro ingresado y envía el mensaje de confirmación, disminuye en uno el número de clientes.

4. Buscar registro:

Cliente:

Crea una estructura dogType y le asigna espacio en memoria, pide el nombre de la mascota, crea una variable cadena 'nombre' y le asigna espacio en

memoria, envía el nombre al servidor, recibe del servidor el id, si id es igual a -5 imprime que no hay resultados, si es igual a -4 hace break, si id es otro número recibe la mascota y la imprime, hace esto hasta que encuentre todos los registros con en el nombre indicado, recibe el mensaje del servidor y lo imprime, se pide al usuario oprimir cualquier tecla para continuar.

Servidor:

Crea una variable cadena 'nombre' y le asigna espacio en memoria, recibe el nombre de la mascota del cliente, busca en la tabla hash desde 0 el nombre indicado, y lo envía al cliente, guarda en el archivo serverDogs.log la información del registro ingresado y envía el mensaje de confirmación, , disminuye en uno el número de clientes.

5. Salir

Cliente:

Recibe el mensaje del servidor y lo imprime, rompe el proceso.

Servidor:

Envía el mensaje de despedida al cliente, cambia el tamaño de la base de datos por el número de mascotas y guarda la tabla hash, sale del proceso, cierra el cliente actual, pone disponible el hilo, disminuye en uno el número de clientes.

Funciones Implementadas En el Código Fuente:

char * toLower(char *arr):
Recibe un puntero a un carácter (por lo general el identificador de un arreglo) y transforma la cadena a los caracteres en minúscula y retorna el apuntador a la cadena en minúscula.

char getch():
Cuando se invoca la función el proceso espera hasta que se presione alguna tecla para continuar con su ejecución.

void ImprimirEstructura(struct dogType * mascota):
Esta función recibe como parámetro un apuntador a la estructura dogType relacionada con una mascota e imprime los valores de la estructura en la consola.

struct dogType * LeerEstructura():
Reserva un espacio de memoria para almacenar una estructura dogType y solicita sus campos por consola, retorna el apuntador a la estructura dogType.

FILE * bdLectura():
Retorna un apuntador al archivo dataDogs.dat para el modo lectura.

FILE * bdEscritura():
Retorna un apuntador al archivo dataDogs.dat para el modo escritura al final del archivo.

FILE *bdActualizar():
Retorna un apuntador al archivo dataDogs.dat para el modo lectura y actualización.

FILE * TablaHashEscritura():
Retorna un apuntador al archivo tabla.dat que almacena la tabla hash, para el modo escritura.

FILE * TablaHashLectura():
Retorna un apuntador al archivo tabla.dat que almacena la tabla hash, para el modo lectura.

void cargarTablaHash():
Lee la tabla hash del archivo tabla.dat y la almacena en el arreglo global tablahash.

int hash(char *str):

Recibe un puntero de tipo char relacionado con una cadena de texto y retorna un entero que representa la función hash que permite ubicar el registro en la tabla.

`void moverPunteroBd(FILE ** puntero,int posiciones)`:
Recibe un puntero doble a un archivo para de esta manera mover el puntero cierta cantidad de posiciones (del tamaño de la estructura dogType) en el archivo para realizar lectura o actualización de registros.

`void ImprimirNodo(struct Nodo *nodo)`:
Imprime el entero "anterior" que designa la posición del anterior elemento de la lista,imprime el entero "siguiente" que designa la posición del siguiente elemento de la lista e imprime los campos de la estructura almacenada en el nodo actual.

`void ActualizarMascotaenBD(int posicion,struct Nodo * nodo)`:
Recibe la posición de la mascota en el archivo y reemplaza la estructura en la posición especificada por la estructura dogType almacenada en la estructura Nodo pasada como argumento.

`struct Nodo *LeerdeBD(int posicion)`:
Retorna el nodo almacenado en el archivo dataDogs.dat en la posición pasada como argumento.

`void InsertarMascotaenBD(struct dogType * mascota,int indice_del_anterior_en_lista)`:
Toma la estructura a la que apunta *mascota y la pone al final del archivo dataDogs.dat encadenándola con la mascota que se encuentra en el índice especificado por el segundo argumento.

`int indiceTaildelaLista(int indice_front)`:
Retorna la posicion del ultimo elemento de la lista y recibe como argumento la posición del elemento del frente de la lista.

`int  Colisiona(char *nombre_nuevo,int posicion)`:
La función compara si el nombre nuevo colisiona con el nombre que se encuentra en la posición especificada por el segundo argumento en el archivo dataDogs.dat. La función retorna 1 si hay colisión y -1 si no la hay.

`int sondeo(int i)`:
Retorna el cuadrado de i, esta función es útil para los casos en los que se presentan colisiones y se realiza un sondeo para encontrar una nueva posición para insertar o buscar un nombre.

void GuardarMascota(struct dogType * mascota,int i):
Guarda la mascota apuntada por el primer argumento con su respectivo hash y sondeo dado por el parámetro i

void CambiarTamanioBd(int tamanio):
Actualiza el tamaño de la BD en el archivo tambd.dat y lo reemplaza por el valor del parámetro.
void GuardarTablaHash():
Toma la tabla hash en ejecución en el proceso y lo almacena en el archivo tabla.dat
char* concat(char *s1, char *s2):
Concatena dos cadenas y retorna un apuntador a la nueva cadena.
void BuscarPorNombre(char *nombre,int i):
La función imprime la lista de registros que tienen coincidencia con el nombre especificado, el parámetro i indica el sondeo aplicado.
int indiceEnTablaHash(char * nombre):
Retorna la posición en el archivo dataDogsdat.dat del último elemento de la lista de registros que coinciden con el nombre pasado como parámetro.
void DesconectarMascota(int posicion):
Suprime de la lista en la que se encuentre a la mascota de la posición pasada como parámetro.
void EliminarDeLaBD(int posicion):
Elimina la estructura que se encuentra en la posición especificada y reindexa el archivo dataDogs.dat y la tablahash.
void EliminarMascota(int posicion):
Llama a las funciones DesconectarMascota y EliminarDeLaBD y disminuye el número de mascotas.
void BuscarPorNombreyEnviaraCliente(char *nombre,int i,int clientfd):
Busca las estructuras que tengan un nombre que coincida con el buscado y lo envían al cliente dado por el descriptor del tercer argumento, el parámetro i es para sondeo.
void EnviarArchivo(int descriptor,char * filename):
Permite enviar el archivo del segundo argumento al cliente del descriptor caracter por caracter.
void RecibirArchivo(int descriptor,char * filename)

Permite recibir el archivo del segundo argumento al cliente del descriptor caracter por caracter.

Diagrama General de Comunicaciones:

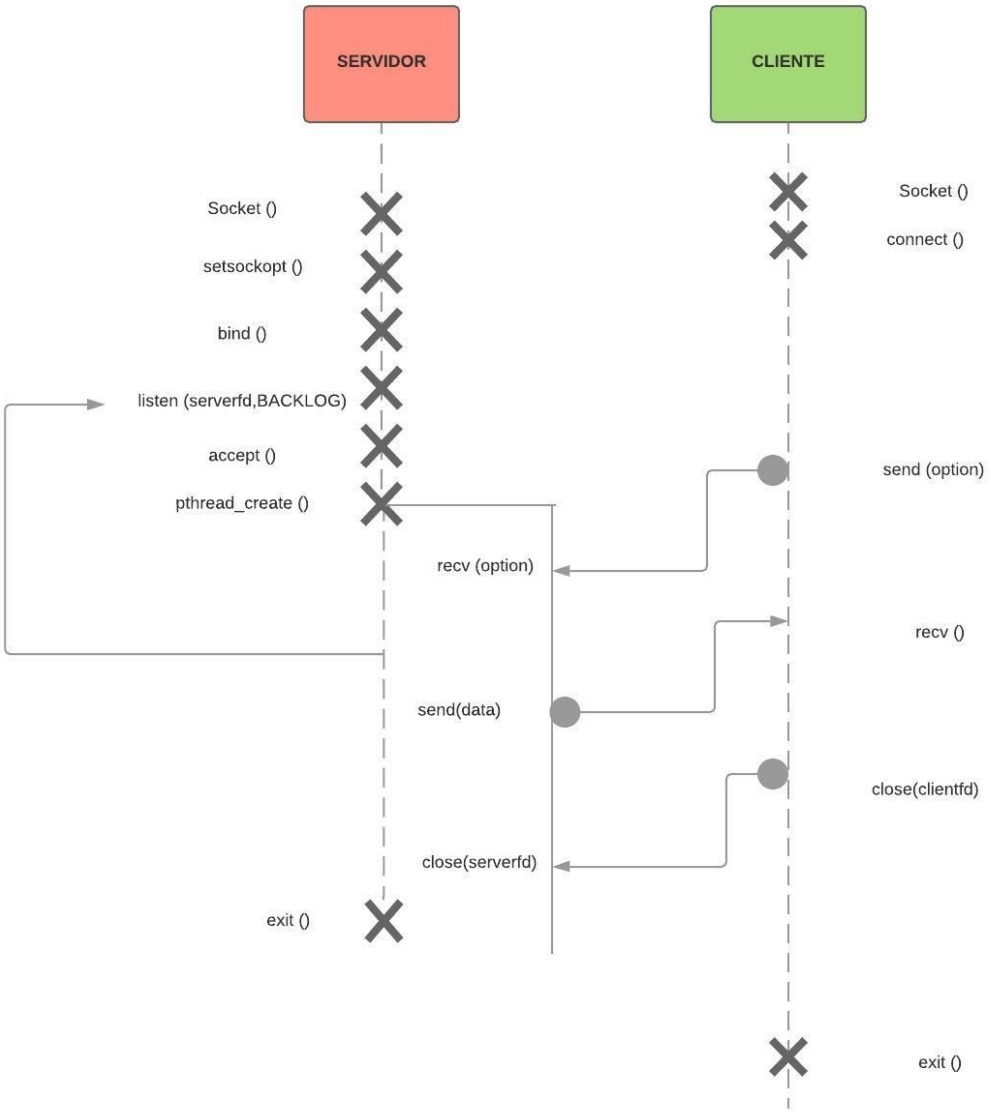
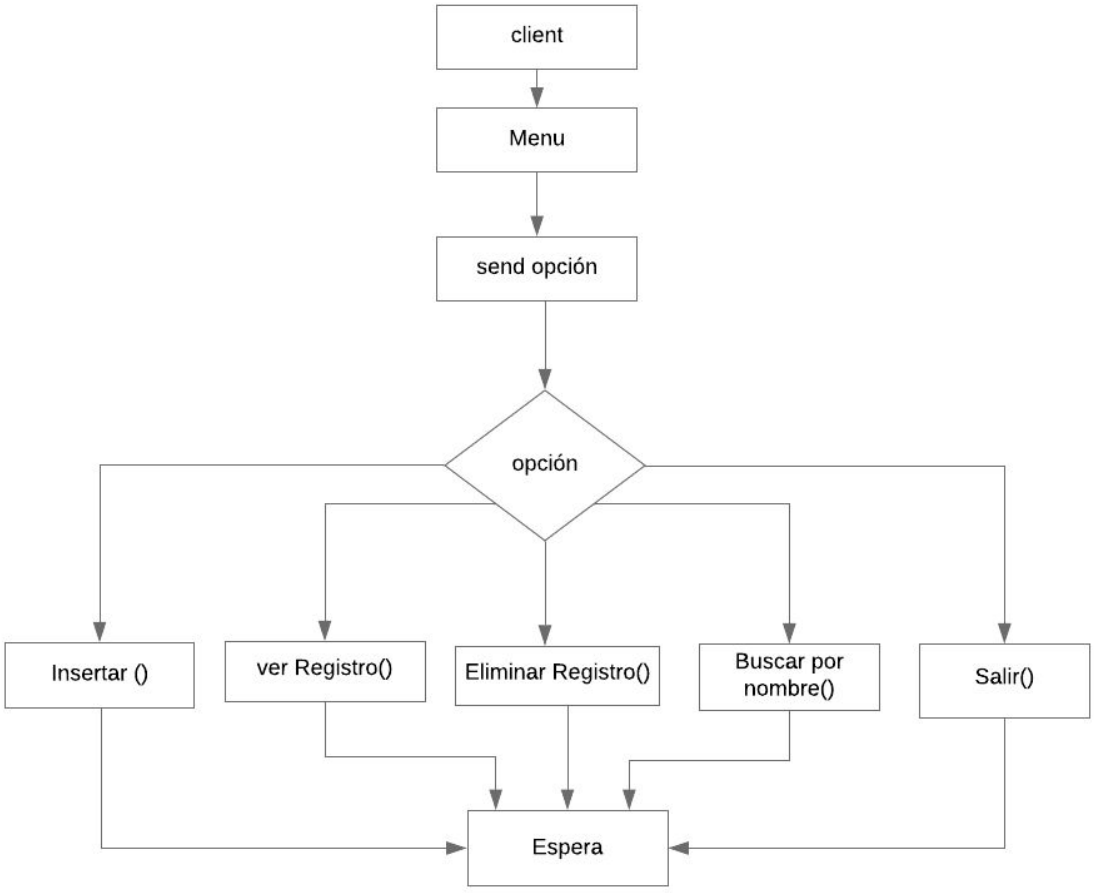
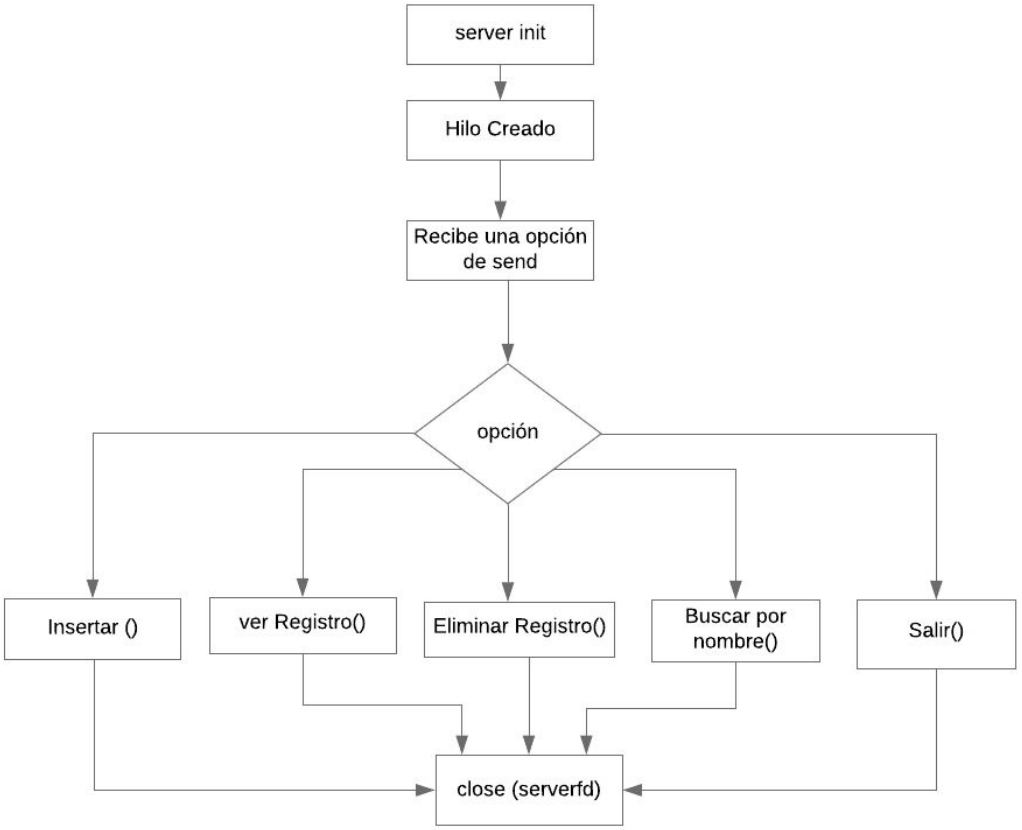


Diagrama de Bloques:

Servidor:


client
Menu
send opción
opción
Insertar ()
ver Registro()
Eliminar Registro()
Buscar por nombre()
Salir()
Espera

Cliente


server init
Hilo Creado
Recibe una opción de send
opción
Insertar ()
ver Registro()
Eliminar Registro()
Buscar por nombre()
Salir()
close (serverfd)