Assignment 5 Report

Futures: Part 2

In this assignment, we added two future states FUTURE\_SHARED and FUTURE\_QUEUE.

I created future queue structure in f\_queue.h

typedef struct f\_queue

{

pid32 process\_id;

struct f\_queue\* qnext;

}f\_queue;

Here f\_queue will store the process id and pointer to next f\_queue node.

Implementation of future\_queue is in f\_queue.c

void f\_enqueue (f\_queue\*\* queue, pid32 pid)

This function is used for queuing the process in queue.

pid32 f\_dequeue(f\_queue\*\* queue)

This function is used to remove the process id from the queue.

Queue functions are implemented as FIFO.

Future\_alloc() is used to allocate the memory to future. In this function, get\_queue and set\_queue are set as NULL.

future\_free() is used to free the future. If get\_queue and set\_queue are not empty then all processes in these queues are killed.

Future\_get function returns the value stored in the future. In case of FUTURE\_SHARED, if future\_state is FUTURE\_VALID then Future\_get returns the value present in future. If future\_state is FUTURE\_EMPTY, then requesting process is stored in get\_queue and that process is suspended. When future\_state becomes valid, all processes are resumed, and future value is returned. In case of FUTURE\_QUEUE, if there is a process present in set\_queue, that process is resumed. Future stores the value set by recently released process. This value is returned to the process which invoked future\_get function. If there is no process present in set\_queue, requesting process is stored in get\_queue and that process is suspended.

Future\_set function sets the value. In case of FUTURE\_SHARED, if future state is FUTURE\_EMPTY then value of future is set. If Future\_state is FUTURE\_WAITING, value of future is set, and processes present in get\_queue are resumed. If future\_state is FUTURE\_VALID, then error is returned. In case of FUTURE\_QUEUE, if there is a process present in get\_queue, that process is resumed. Future stores the value set by calling process. This value is returned to the released process. If there is no process present in get\_queue, requesting process is stored in set\_queue and that process is suspended