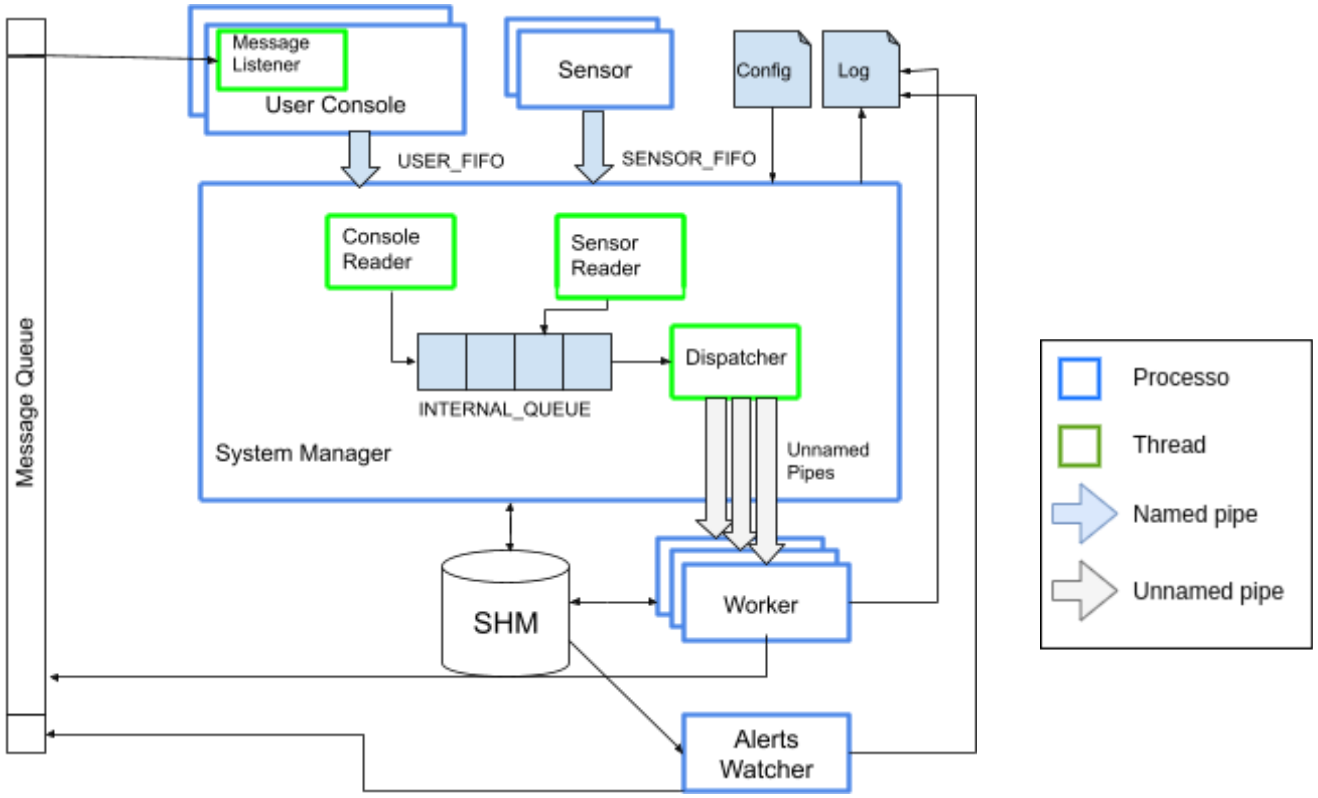
# Diagrama:




# Relatório:

User Console:

- Possui a thread (message_listener) que ouve a Message Queue e imprime o seu conteúdo;
- Envia os comandos para o sys_manager através de um pipe (verificando os atributos), guardando tudo em uma struct “Command”;
- Sinais:
    - **SIGINT -** Dá cleanup a memória alocada e fecha a consola.

Sensor:

- Gera um valor random entre “min_value” e “max_value” a cada “inter” segundos (esses valores são definidos ao criar o sensor, sendo estes parâmetros verificados);
- Sinais:
    - **SIGTSTP -** Mostra o número de mensagens que já foram enviadas pelo pipe até o momento da chamada.
    - **SIGINT -** Chama a função para dar close ao pipe, mostrando o número de mensagens enviadas no total.
    - **SIGPIPE -** Chama-se uma função que desconecta do pipe com segurança e mostra o número de mensagens enviadas no total. (só é ativado quando tenta escrever para um pipe que está fechado).

Filipe Rodrigues - 2021218054
Joás Silva - 2021226149

SysManager:

- Console Reader:
    - Usa o pipe USER_FIFO para se comunicar com o processo user_console;
    - Cria um Job com tipo 1 (do usuário), lê o comando pelo pipe, envia o Job para a internal (utilizamos o mutex BLOCK_QUEUE para só poder adicionar a queue quando houver espaço, também utilizamos a variável de condição "PRIO_QUEUE" para dar prioridade a adição na Internal Queue [quando esta estiver cheia]);
- Sensor Reader:
    - Usa o pipe SENSOR_FIFO para se comunicar com o processo Sensor;
    - Cria um job do tipo 0 (do sensor);
    - Envia o Job para a Internal Queue com a ajuda do mutex BLOCK_QUEUE;
    - O Job é descartado se a fila estiver cheia;
- Dispatcher:
    - Lê um Job (espera até que a Internal Queue esteja disponível utilizando a variável de condição COND_QUEUE);
    - Remove o Job da queue e guarda-o em uma variável (notificando a variável de condição PRIO_QUEUE);
    - Da wait ao semáforo MUTEX_JOB até encontrar um worker disponível, também utiliza o semáforo DISP para regular o acesso aos workers na memória partilhada;
    - Após encontrar um worker disponível, define-o como ocupado
- Worker:
    - Fica a espera até receber um Job (fica a ler a Pipe, sem busy wait);
    - Após isso, bloqueia o Sighandler para o processo não fechar enquanto estiver a executar um trabalho;
    - Executa o trabalho desejado (da User Console ou do Sensor);
    - Se o Job foi enviado por uma User Console, envia a resposta para a Message Queue;
- Alert Watcher:
    - Da wait ao semáforo WAIT_ALERT (que só recebe post quando um sensor envia uma mensagem);
    - Percorre todos os alertas para verificar se algum foi acionado (apenas verifica os alertas correspondentes ao sensor que enviou a mensagem);
    - Caso sim, envia uma mensagem para a messsage queue, endereçada a User Console que definiu o alerta;
- Sinais:
    - **SIGINT -** Chama função para limpar todos os recursos utilizados.
    - **SIGUSR1 e SIGUSR2 -** Sinais para dar handle ao exit das threads e processos, respetivamente.

Filipe Rodrigues - 2021218054
Joás Silva - 2021226149