Prueba para Desarrollador C++

Crear un servicio/demonio/agente, que debe ejecutar tareas a demanda.
Las tareas deben ser recibidas de forma externa, sea usando un archivo externo, desde otro proceso (Se puede usar boost::interprocess), etc.
Todas las tareas deben tener una hora en la que será ejecutada.
Las horas deben estar en formato Unixtime.
El resultado de las tareas debe ser escrito en la base de datos.
Si alguna tarea debe mostrar información se debe guardar en un archivo en formato JSON

Formato de las tareas en JSON:

```
{
  "task”:integer,
  "hour”:number,
  "detail”:string
}
```

Tareas que debe ejecutar el programa:

Tarea 1:
Extraer la información del procesador (Nombre, número de núcleos, hilos, caches, etc) usando las API del Sistema Operativo
Bonus: Extraer información de la Memoria (Espacio ocupado, espacio libre, modelo, tamaño, serial, etc)

Ejemplo de petición:

```
{
  "task”:1,
  "hour”:1593565478
}
```

Tarea 2:
Buscar un archivo en el equipo y guardar la información relacionada, tamaño, ultima modificación, etc.
Bonus: Puede recibir patrones, ej: *.jpg

Ejemplo de petición:

```
{
  “task”:2
  “hour”:1593565478
  “detail”:”file.txt”
}
```

Tarea 3:
Consultar toda la información contenida en la base de datos

Ejemplo de petición:

```
{
  "task”:3,
  "hour”:1593565478
}
```

Bonus general:
Agregar soporte compilación Clang en las 3 plataformas configurable desde Cmake
Agregar pruebas unitarias con GTest o Catch

Requisitos:

- Usar hasta C++17
- En MacOS se puede usar donde se requiera Objective C
- Usar Git
- Subir proyecto a un repositorio
- Multiplataforma Linux (GCC), Windows (MSVC), MacOS (Clang)
- Usar CMake
- Los proyectos se deben poder compilar en cualquier equipo
- Todas las bibliotecas de terceros deben ser compiladas de cero y agregados al proyecto
- Para la conexión de base de datos se debe usar SQLite a través de SOCI
- Agregar un README con las instrucciones