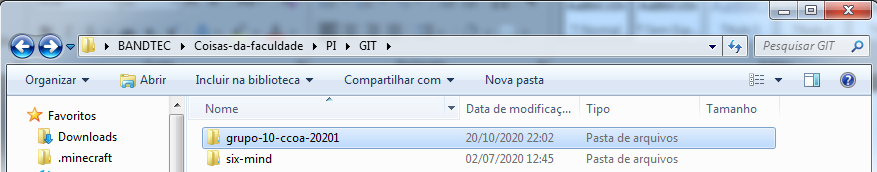
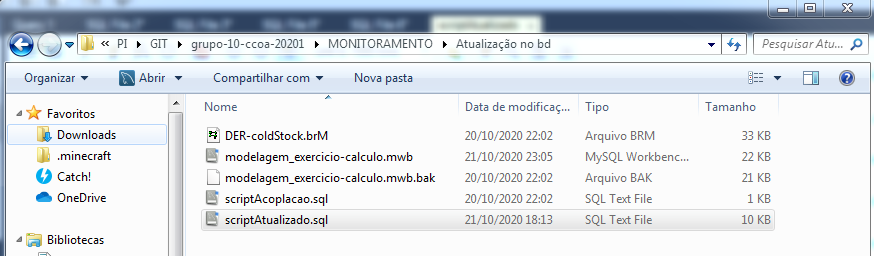
Relatório das API’s

Primeiramente vamos ao GitHub e vamos clonar o repositório das nossas API:



Depois disso precisamos criar as tabelas no nosso banco de dados para que possamos inserir e visualizar os dados que iremos gerar. Dentro do nosso repositório temos o script que cria todas as tabelas que precisamos então vamos até o repositório onde está nosso script e rodamos ele no Mysql Workbench:

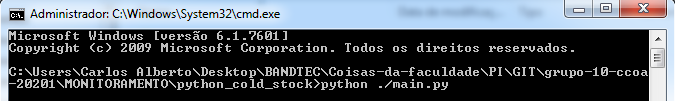


Interface gráfica do usuário, Texto, Aplicativo

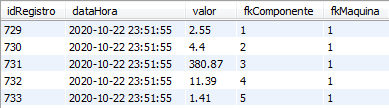
Descrição gerada automaticamente

Com todas as nossas tabelas já criadas vamos para a parte de rodar nossas API’s. Nós estamos utilizando duas API’s, uma utiliza os comandos psutil, e a outra API é um web crawler que utiliza o OpenHardwareMonitor para capturar os dados da nossa máquina.

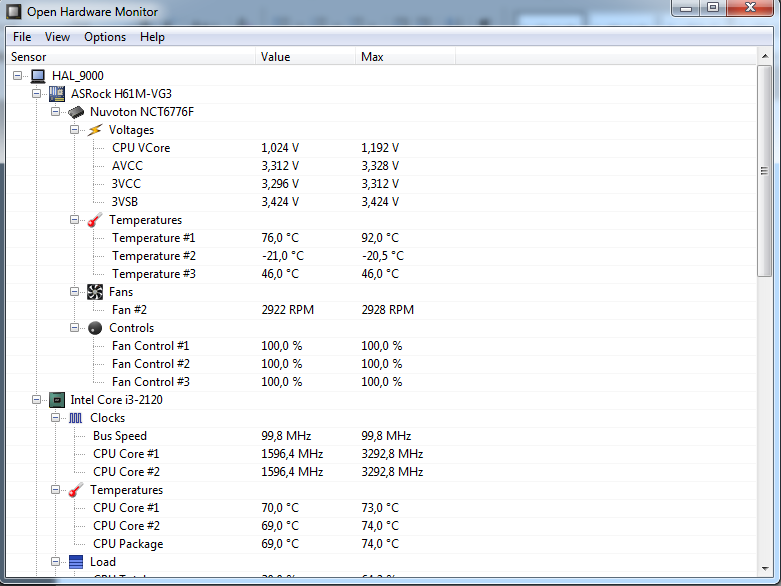
Api pythonycs: Para nós rodarmos nossa api pythonycs basicamente entramos na pasta python\_cold\_stock com o nosso cmd e inserimos o comando: “python ./main.py”



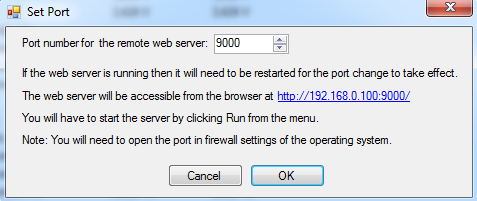
Com esse comando a nossa API já faz os inserts dos dados capturados por comando da biblioteca psutil, esses dados são de memória, disco, conexão de dowload e upload



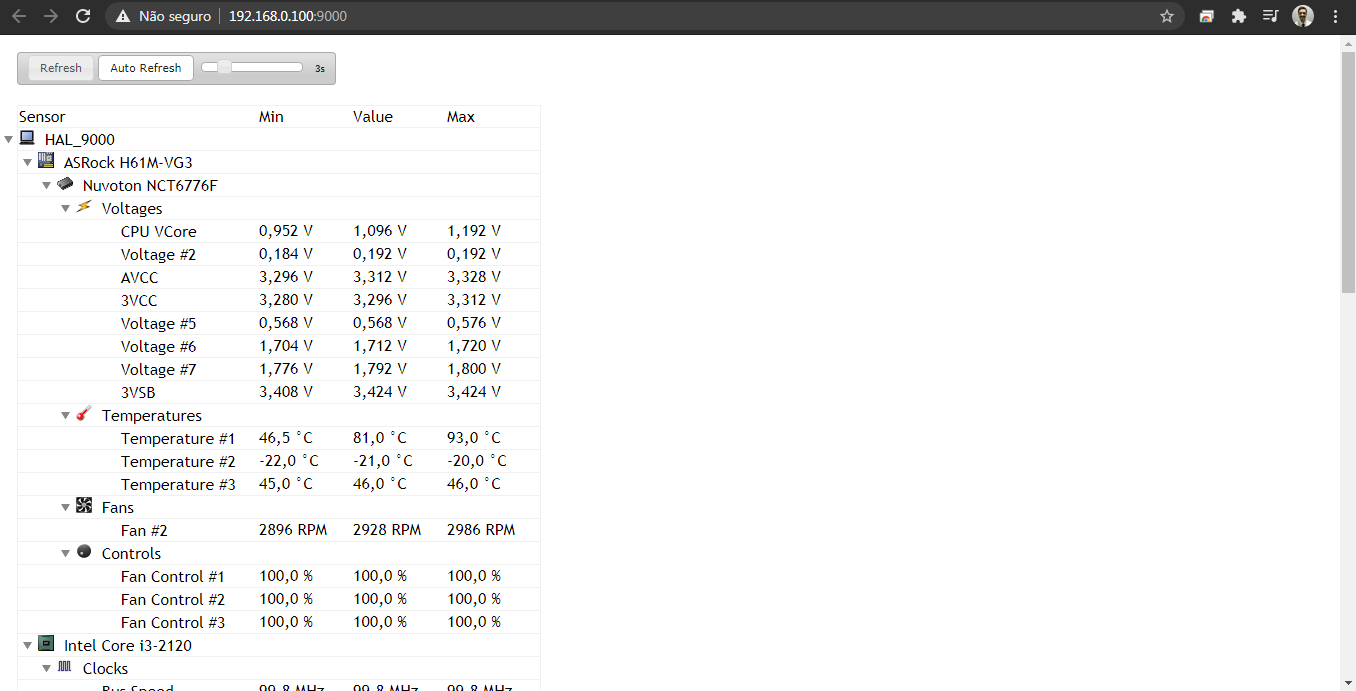
Nossa aplicação utiliza uma tabela modular, ou seja, não fizemos uma coluna para cada componente analisado e sim uma coluna com todos os valores captados e com um fkComponente indicando a qual componente aquele valor se refere.

API pythOHNs: Para rodar nossa API pythonhs temos um caminho diferente, primeiramente abrimos o aplicativo do open hardware monitor 

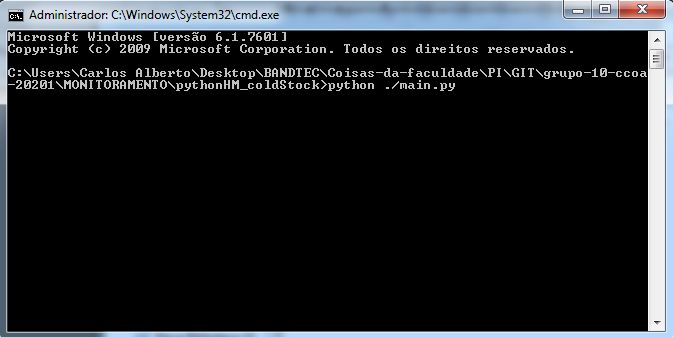
No aplicativo vamos em Options -> Remote Web Server -> run, assim o aplicativo roda no nosso localHost e setamos para que usamos a porta 9000

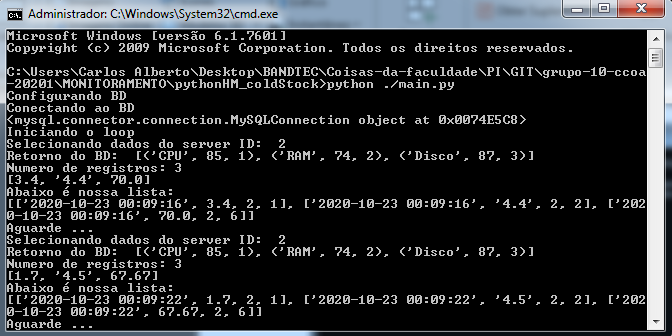


Então vamos até o nosso navegador e entramos no endereço <http://192.168.0.100:9000/> e assim os dados são enviados ao nosso navegador e assim a API pode funcionar como um Web Crawler e capturar os dados do navegador em si

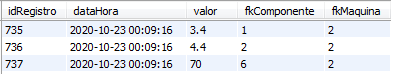


Após esse passo nós apenas precisamos entrar na pasta pythonHM\_coldStock com o cmd e executar o comando: “python ./main.py”





A API pythohns capta dados de frequência de CPU, RAM e a temperatura do CPU, e faz insert desses dados no mesmo formato da outra API citada acima, então por isso utilizamos a mesma tabela no banco de dados e referenciamos os dados dos valores com campo fkComponente



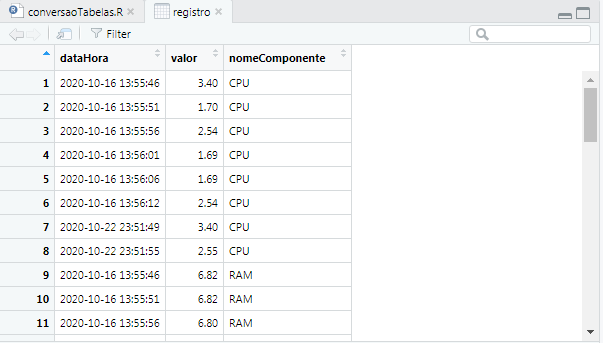
Nossas duas API’s estão rodando e fazendo os inserts de forma correta, então agora vamos fazer a modelagem matemática dos dados utilizando a ferramenta R studio, então primeiramente vamos exportar a tabela registros para que possamos modelar esse dados e simular uma máquina servidor por exemplo

Importamos a tabela registro do nosso banco de dados no formato .csv utilizando o Table Data Export Wizard com o seguinte comando select:

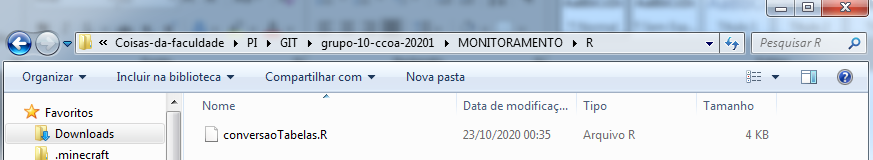
“select dataHora, valor, nomeComponente from coldstock.registros INNER JOIN coldstock.componentes on fkComponente = idComponente where fkMaquina = 1 order by dataHora;”

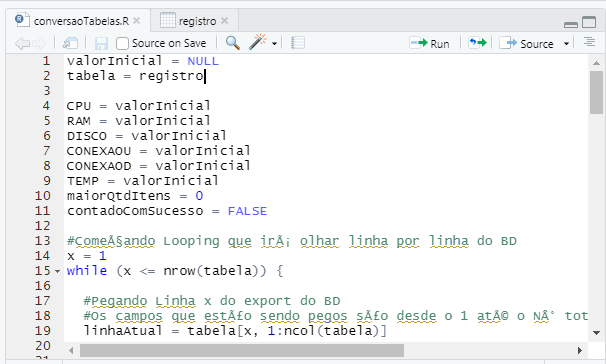
Após exportarmos essa tabla que chamamos de “registro.csv” importamos ela para o R



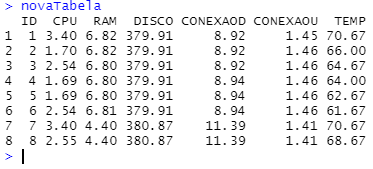


Então voltamos ao diretório do nosso projeto e vamos atrás do script que criamos para alterar nossa tabela de forma com que possamos modelá-la de forma mais simples. Já que fizemos uma tabela modular modificamos ela com um script em R para que cada componente tenha um coluna específica





Assim criando essa rodando esse script criamos uma tabela chamada “novaTabela” e apartir dela modelamos de forma simples e fácil os dados capturados



No exemplo abaixo vamos simular o CPU de um servidor multiplicando os valores capturado por 3, assim sendo um servidor 3 vezes mais potente que nossa máquina

