OS project report

Seifeldin Elshabshiri
Amr Hussein
Moneer Maged Zaki

May 2023

1 Abstract

In this report we will talk about CLI and GUI, and discuss our implemeted features. We implemented a task manager that supports both a gui and a cli, then defined its design. We named Task manager "SAM", and we desired to build it with rust as it is a bonus.

2 Introduction

In this paper, we are talking about our task manager "SAM". So, after lots of extensive research, we began to choose the features that we will implement. We built our task manager using CLI task manager with GUI component. We built the GUI component from scratch but for CLI, we gain aid from tkm task manager and integrate our features on it.

3 Member roles

3.1 Amr Hussein

- CLI Process Information: change priority
- CLI System Information: overall CPU
- GUI Process Information: Graphical representation for the process information.
- GUI System Information: CPU usage using Pie Chart
- GUI System Information: Memory usage using Pie Chart
- Graphics for GUI representation: UI/UX implementation for the resulting GUI.

- Graphics for GUI representation: Using HTML, CSS, JS to represent the data produced by RUST files.

3.2 Seif Elshabshiri

- CLI Process Information: Sort by "nice" and "ppid".
- CLI Process Information: Filter by all possible columns including: "pid, ppid, state, nice(priority)". note: name is filtered using regex().
- CLI Process Information: Adding state, "nice", "ppid", "columns" to the process table
- CLI Process Information: Making the table columns customizable.
- CLI System Information: overall memory
- CLI General Information: Help function to display all commands that the user can choose from.

3.3 Moneer Zaki

- Project Management
- CLI Process Information: Detecting processes using colors.
- CLI Process Information: regex() function that matches and filters the name of the process that is entered by the user.
- GUI Process Information: Same information as the CLI process manager.
- GUI System Information: Tabular representations for histories.
- Graphics for GUI representation: JS code edit for easier readability.

4 Resources Credits

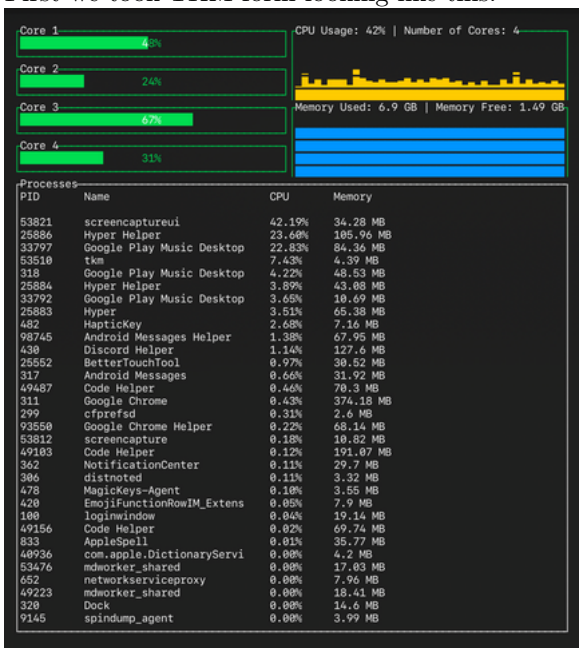
We used tkm Task manager from Github.

5 Description for both CLI and GUI with Screenshots

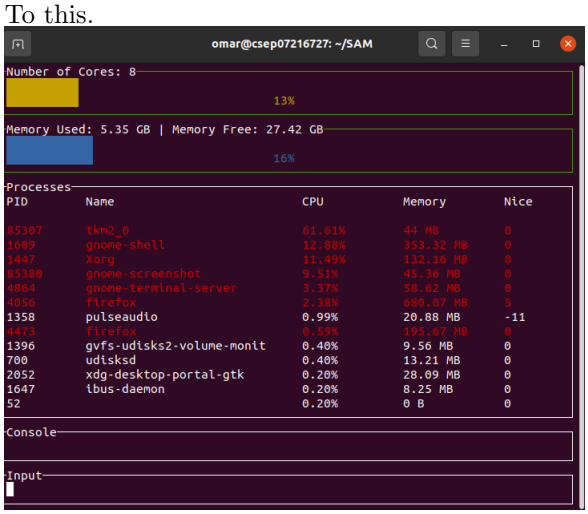
5.1 CLI

Our main focus in this project was the CLI. For this we used TKM by SammperlMuter on github. This was an open source project. We mainly used it for their easy to use rendering of components and their file management in their project.

However, we saw that it lacked some of core functionalities that a task manager needs, we also felt like their user interface was cluttered by unnecisary data, that doesn't really offer any value to the user.
First we took TKM form looking like this.



To this.



We then added some functionalities that TKM was lacking. We added more columns to each process. To implement this without cluttering the interface, we fixed two of the columns, PID and Name, and made three rotatable rows that fit 5 columns. The user can do the command 'add state' for example, and the interface will change to have PID, Name, Memory, Nice, and State as

the columns, adding state to the right and removing cpu from the left. This function can be done for all 5 columns. We used the ready made input and console functionalities but made it show on the screen to the user, and added functionalities that the user can input.

Another functionality that we added is colored outputs, in which processes are colored red if they have high cpu or memory usage

We also added that the user can filter by name, using regex wildcards, or filter by pid, state, ppid, or nice. The user can also use a combination of filters and remove all filters by pressing ctrl + r.

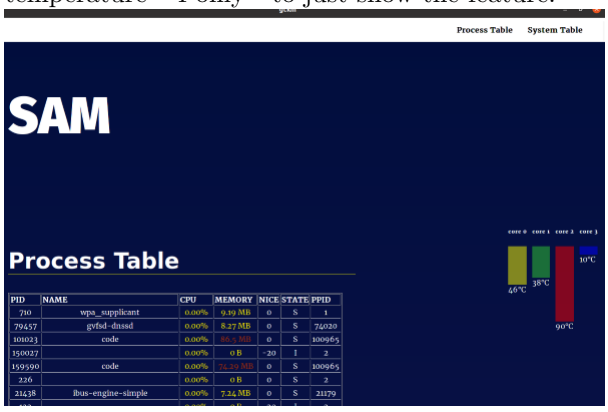
Additionally we added a decrease priority and increase priority of any process using its PID.

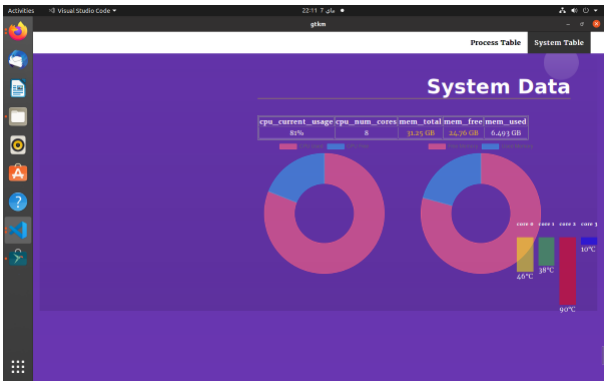
We also changed the graphical output to only include a gauge for memory usage and history usage.

Lastly the user can run a command on the linux terminal, in order to see all the possible commands described above.

5.2 GUI

We implemented it with navigation bar that has 2 component: Processes table, and System table, by clicking on each of them you will see the content of it. Then, for processes table, when the cursor stop at any row, it highlights it. For System table, we implemented 2 charts, by clicking on any of its label you can either appear of disappear it. We also present bars to represent each core's temperature - 4 only - to just show the feature.





6 documentation

- SAM Features
  - CLI representation for Processes and their PID, PPID, Name, CPU Usage, Memory Usage, Nice(priority), and State.
  - CLI representation for System information, including: cpu usage, memory usage.
  - GUI representation for Process in a tabular form.
  - GUI representation for System information in a Graphical and a user friendly way.
- Additional Features
  - GUI for the data representation.
  - Graphs, Pie charts and tabular representation is used.
  - Sorting and filtering with more table columns than already presented by TKM in CLI.
  - Searching for process names using regex() function that enables the user to search easier.
  - Changing priority, and then sorting by it.
  - colored output in CLI.
  - Customize the columns of the process table represented in the CLI.
  - Help function that prints all possible commands to the user.
- Bonus Features
  - Rust-Based Task manager.
  - Temperature of the system.
  -

7 Thanks

Great Thanks for Dr. Amr Elkady for Giving us the opportunity to go through such great experience and make our own Task manager.