Design document

Case Study

Robert Ašvicas

Dmitry Lvov

Tutor:

Mikaeil Shaghelani

Date:

2020-04-21

**Contents**

[Introduction 3](#_Toc38829471)

[Minimal requirements 3](#_Toc38829472)

[Desirable outcome 3](#_Toc38829473)

[Roles and actors 3](#_Toc38829474)

[MoSCoW 4](#_Toc38829475)

[Infrastructure of our environment 5](#_Toc38829476)

[Virtual Machines configuration table 6](#_Toc38829477)

[System setup 7](#_Toc38829478)

[Virtual Machines 7](#_Toc38829479)

[Windows Active Directory 7](#_Toc38829480)

[Windows Client Machine 7](#_Toc38829481)

[Linux Server 7](#_Toc38829482)

[Firewall 7](#_Toc38829483)

[Group policies 8](#_Toc38829484)

[VPN 9](#_Toc38829485)

[Folder Redirection 9](#_Toc38829486)

[Backup Policy 9](#_Toc38829487)

[Application 10](#_Toc38829488)

[Wireframe 10](#_Toc38829489)

[Description 12](#_Toc38829490)

[Database 16](#_Toc38829491)

# Introduction

The company “Make IT Work4U” wants to mimic IT infrastructure of a customer. Our job is to setup the whole environment for that. That includes infrastructure and application/dashboard to monitor server conditions.

# Minimal requirements

These are the minimal requirements of our project.

* Virtual Servers to simulate environment
  + Windows AD serves the purpose of the Domain Controller: Main Server that handles everything (IP assignment, handles Group Policies, etc.)
  + Windows for client is made to connect to. Mainly will be used for clients to test Group Policies.
  + Linux – for experimental purposes.
  + Firewall – Connects LAN with WAN, furthermore our environment has secure internet access. Plays very big role in our environment.
* Different users and group policies in simulated environment
* Application/dashboard is used to monitor servers
  + Real-time statistics are shown, such as CPU, Memory, Disk usage and Network
  + Database, stores data, helps keeping track of things.
  + Logs are used to keep track of real time stats.

# Desirable outcome

Our desired result of the project is to make environment that is easy-to-use for testing purposes, furthermore clients could easily experiment and see as close as possible to real scenario results. Furthermore, application/dashboard would create logs, databases, and real-time statistics, so it would be easy to keep track of what is happening on the server.

# Roles and actors

**Clients** supposedly will connect to Windows Client using one of few created accounts. They will be able to share files, access internet, but they will not have access to configuring server.

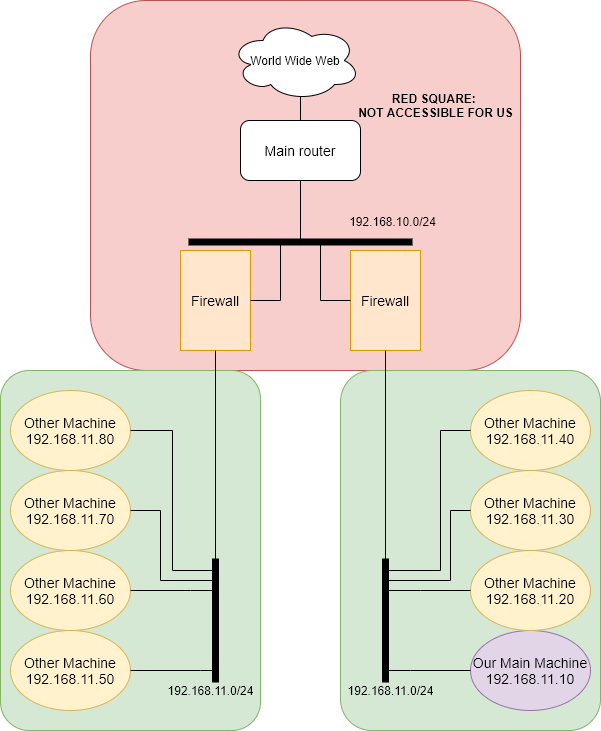
**Administrators** will be able to connect to all machines and configure it how it is needed at that moment. They have full access to all machines.

# MoSCoW

Our prioritization of tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Must have** | **Should have** | **Could have** | **Won’t have** |
| Windows AD server | More users | ADBlock in the Firewall | Backup on external device |
| DHCP server | Advanced User Policies | Proxy with Anti-Virus | IDS |
| DNS server | App: Graphical Diagrams of real time statistics | App: Different color themes |  |
| Firewall | App: In-depth info about users |  |  |
| Proxy |  |  |  |
| VPN |  |  |  |
| Users |  |  |  |
| Group Policies |  |  |  |
| Folder Redirection |  |  |  |
| Application/Dashboard |  |  |  |
| App: Server health stats |  |  |  |
| App: Virtual machine stats |  |  |  |
| App: Log management |  |  |  |
| App: Database tracking |  |  |  |
| Linux Server |  |  |  |
| Backup Policies |  |  |  |

# Infrastructure of our environment



## Virtual Machines configuration table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name in the Hyper-V | Name in the table above | IP address (local) | Netmask | MAC Address | Role |
| Win2019\_AD | Windows Domain Controller | 192.168.1.2 | /24 | 00:15:5D:0B:0C:00 | Active Directory |
| Win2019\_not\_AD | Windows server client | 192.168.1.3-.254 | /24 | 00:15:5D:0B:0C:06 | Client server |
| Linux | Linux Server | 192.168.1.3-.254 | /24 | 00:15:5D:0B:0C:02 | Experimental purposes |
| Firewall | Firewall (pfSense) | 192.168.1.1  WAN IP: (192.168.11.11) | /24 | hn0: 00:15:5D:0B:0C:07  hn1: 00:15:5D:0B:0C:08 | Firewall |
| - | Main Server | 192.168.11.10 | /24 | 00:15:5D:0A:10:07 | Hosts everything mentioned above |

# System setup

One server is hosting 8 Virtual Machines, one of them we will consider as our Main Machine, since we can connect only to it. Our Main Machine is hosting 4 Virtual Machines using Hyper-V, native hypervisor included in Windows Server. On the Hyper-V we have installed Windows with Active Directory (Domain Controller), Windows for clients to connect to, Linux Server (Lubuntu) and Firewall (pfSense). More specific details about each one of them will follow.

# Virtual Machines

All those VMs that are mentioned above are in the same domain name, which is called firstgroup.local. All of them have Dynamic Memory and they are 2nd generations VM, so disk space is also dynamic. They are in the network 192.168.1.0/24.

## Windows Active Directory

Windows Active Directory is responsible for giving out IP addresses (it has DHCP Server installed on it), DNS server, and hosting domain called firstgroup.local. Also, through AD we are managing group policies. This machine has IP 192.168.1.2, netmask 255.255.255.0 and gateway 192.168.1.1, all this is statically configured. In Hyper-V this VM is called “Win2019\_AD”.

## Windows Client Machine

Windows Client Machine is machine made for users to connect to. It shares folders: Documents and Desktop. It is designed mainly for users to connect to it. This machine will be given IP by DC (Domain Controller) (Scope: 192.168.1.3-.254) (most commonly it’s given 192.168.1.3 IP). In Hyper-V this VM is called “Win2019\_not\_AD”.

## Linux Server

Linux server has Lubuntu running on it. It server experimental purposes. In Hyper-V it’s called “Linux”.

## Firewall

pfSense serves as a router and a firewall. It connects 2 networks – WAN and LAN. For that it uses 2 interfaces: hn0 (LAN) and hn1 (WAN) Without firewall firstgroup.local domain won’t have any internet connection. WAN IP: 192.168.11.11/24 – static configuration, gateway 192.168.11.1. LAN IP: 192.168.1.1/24 – static configuration. Firewall forwards DNS queries firstly to DC, then it checks itself, and only then in forwards it outside.

Firewall has Proxy and VPN server enabled on it. In our case proxy is working as a caching server. For VPN information, check page 9.

In Hyper-V, this VM is called “Firewall”.

# Group policies

All of these virtual machines share one domain, furthermore they have same users and group policies. There is a main our made group policy called GTA which applies the same background pictures for user-group SanAndreas which has 2 users, CJ, and Big Smoke. Apart from that we’ve implemented a folder redirection option which communication-wise allows users to share Documents folder.

Groups:

* Administrators
  + Users
    - Admin

Password: Fontys123

* + - Administrator
    - Test

Password: Fontys!123

* + Policies
    - Full system control
    - Folder Redirection
* SanAndreas
  + Users
    - CJ

Password: DamnTrain112233

* + - Big Smoke

Traitor123

* + Policies
    - GTA (Unified background picture)
    - Folder Redirection
* Simpsons
  + Users
    - Homer Simpson

Password: NuclearPlant321

# VPN

Server:

As a VPN server we’ve chosen OpenVPN as it satisfies all our necessities and it is easy to configure and use. The server was configured with default settings which means that the only parameter that we’ve had to change manually was “Tunnel network” 10.0.10.0/24 and DNS domain (firstgroup.local) with servers which are 192.168.1.2 and 127.0.0.1. The server uses basic encryption with self-signed certificate for users who want to connect to it.

Client:

For each user that will use VPN network there will be a separate private account with login and password credentials which they are going to use through Viscosity client to connect to the network. Port redirection allows users to enter the public IP and certain port which will be assigned to them to connect to the network.

# Folder Redirection

Communication-wise all users in the system will use folder redirection function in regards of Documents and Desktop folder. As users remain in the same network, they have possibilities of using the same Documents and Desktop folders which originate from the Windows VM with AD. This option will allow users to exchange information on a higher pace and with better efficiency.

# Backup Policy

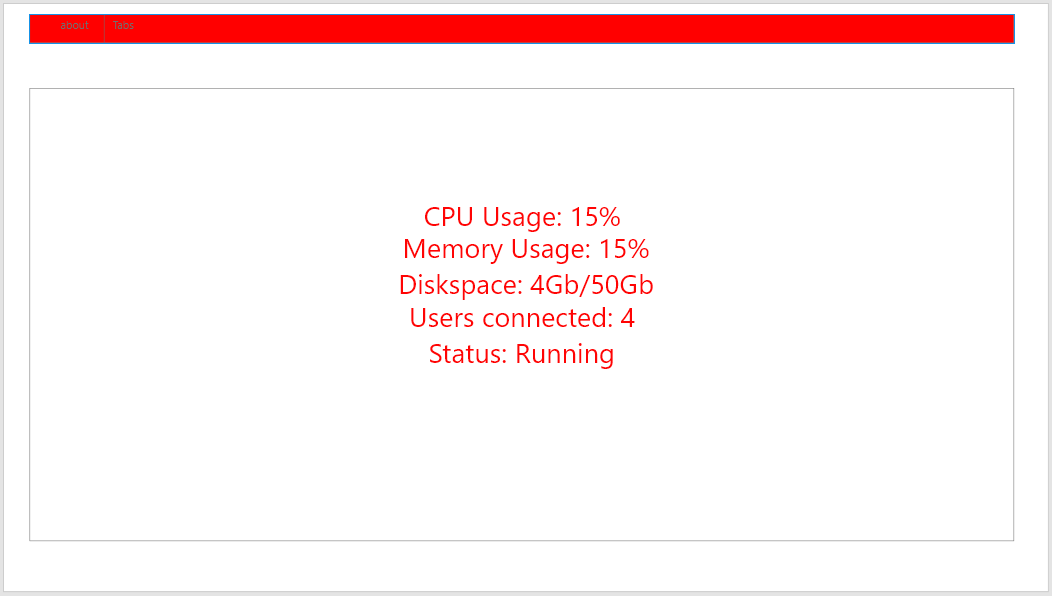
Backups are setup on our Main Machine, it can be accessed via Server Manager -> Tools -> Windows Server Backups. It is backup-ing “Firewall” VM every day at 03:00AM. The backup is stored in shared network folder called “Backup” in Drive C, which is not secure, but right now, it’s the only way to do it (For other ways we have insufficient disk space).

# Application

Application/dashboard is made in Python programming language, which is also installed on our Server. Python version is 3.8.2. For developing GUI we have used Tkinter library , for drawing graphs and statistics matplotlib library and for taking system information psutil library. The information regarding the log files is stored in .txt files on the machine and passed to an SQL database. Real-time statistics are represented in graphs which refresh every second.

## Wireframe

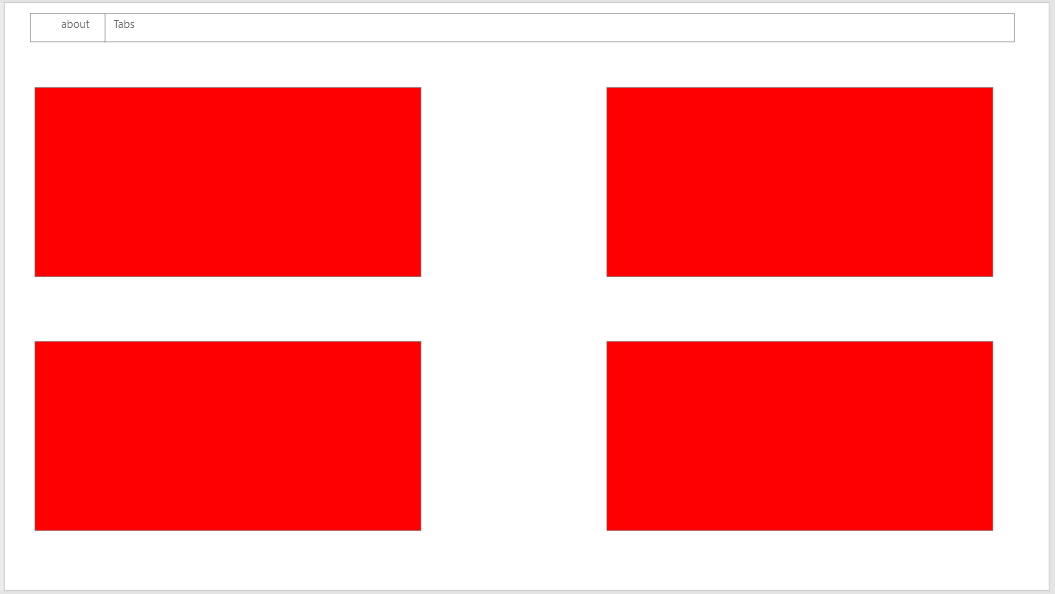
Main page:



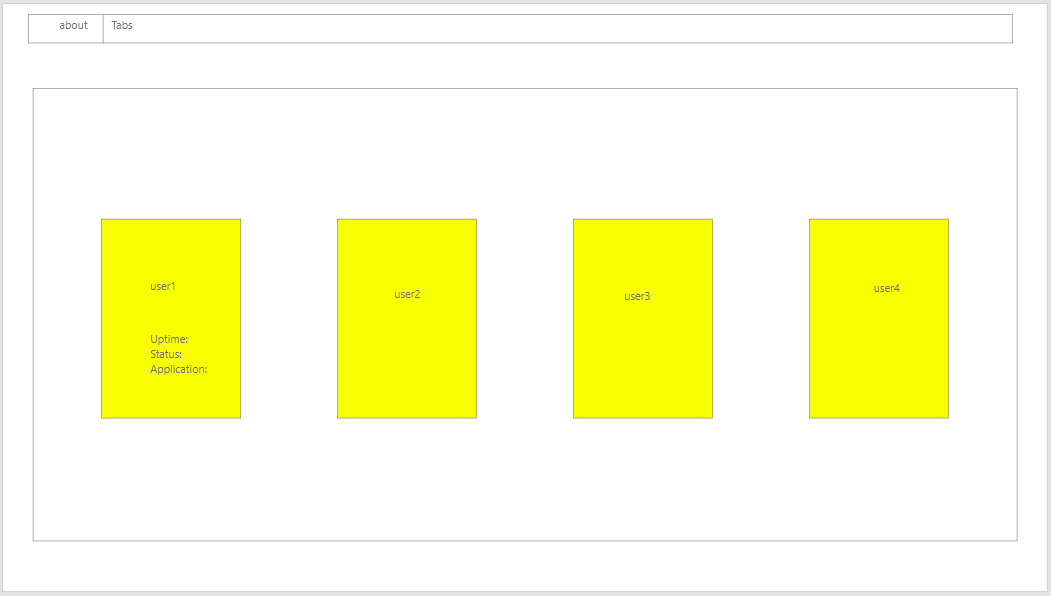
Page 2: Real-Time Statistics



Page 3: Logs



Page 4: Users



Page 5: Processes

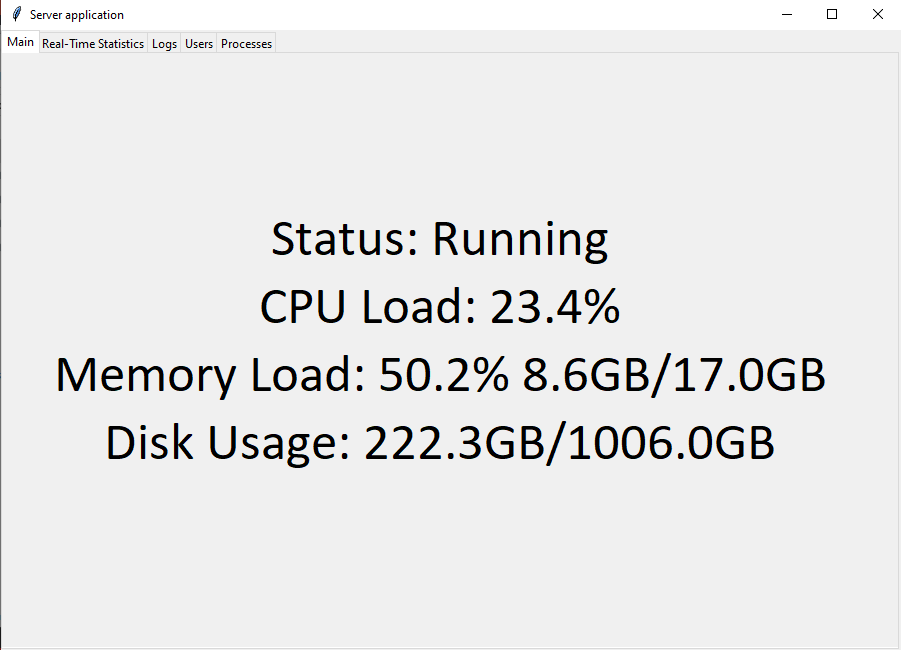


## Description

General:

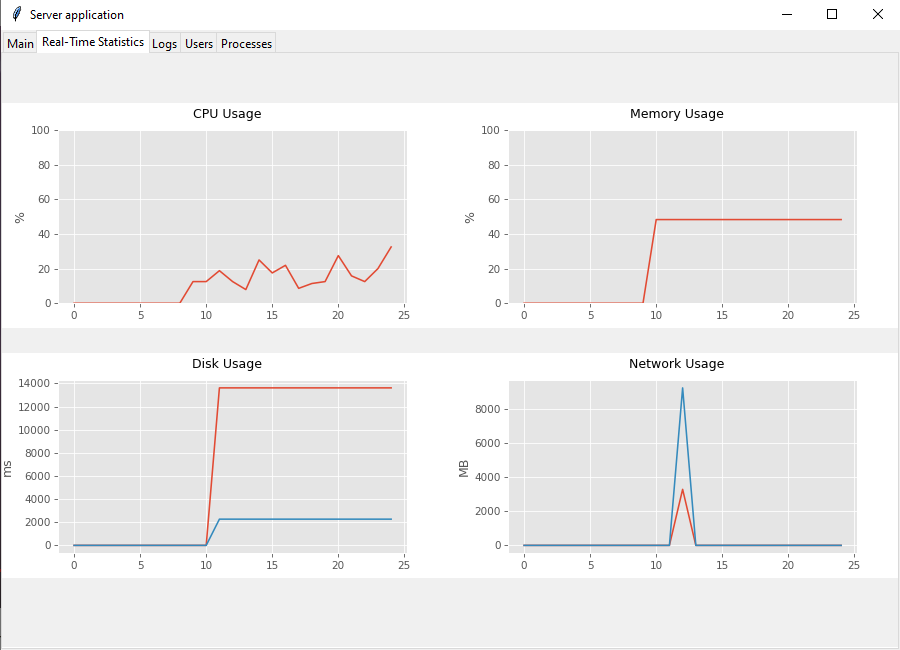
App consists of 5 tabs which contain main information about the machine state.

1. **Main**



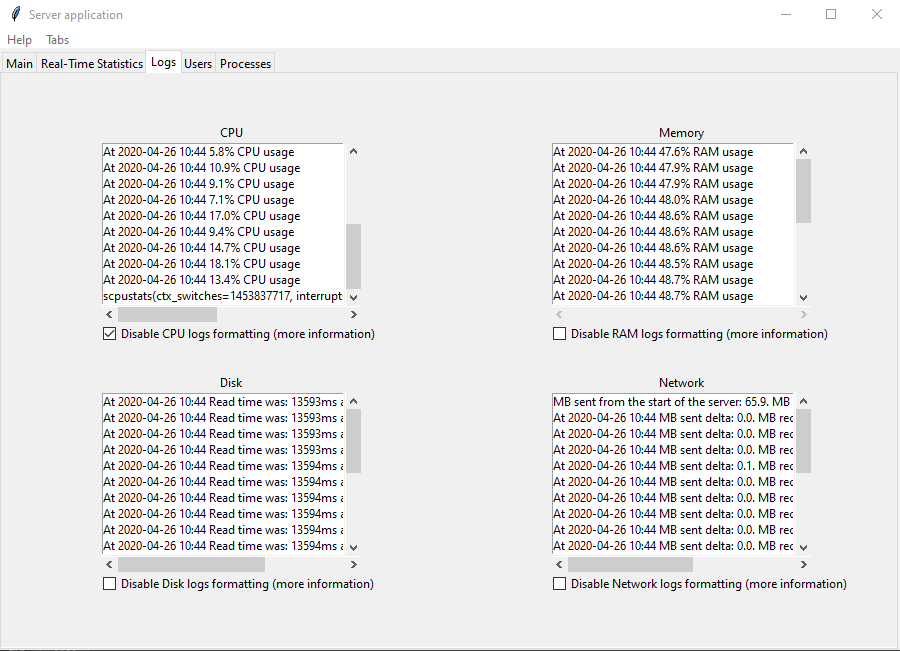
This tab contains 3 labels which are dynamically updated using psutil module which retrieves real-time information about the most significant system parts.

1. **Real-Time Statistics**



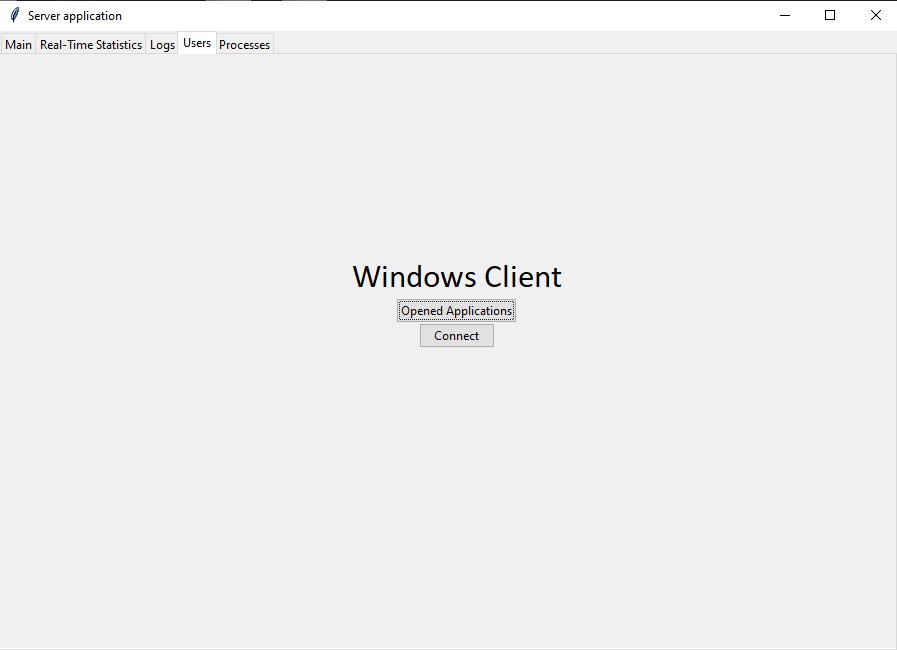
This tab contains 4 graphs made with matplotlib integrated in TTk notebook. They show main information about the system usage and have labels in order to distinguish the processes.

1. **Logs**



The Logs tab consists of 4 listboxes which receive information from psutil library about the health status of the machine. As there are implemented filters to show information in a formated way there are checkboxes which show raw output when activated.

1. **Users**



This tab contains a label with the name of the connected user and 2 buttons:

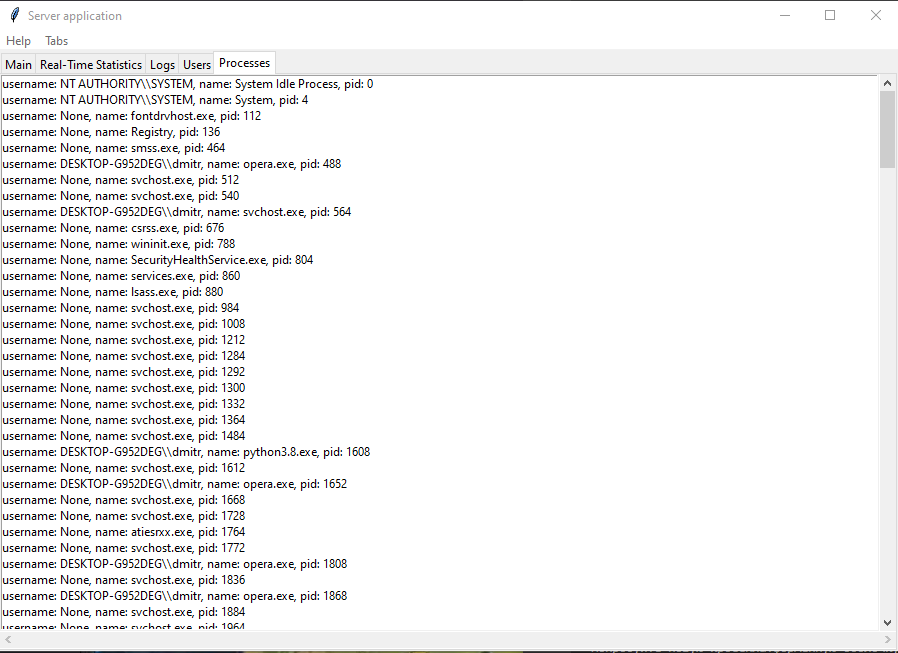
* + Button “Open application”:

This button is responsible for opening a new window for the user which contains a listbox with all processes running on the user machine in a hierarchical order of the importance. The information which passed to the listbox from a file which is created by the “Connect” button

* + Button “Connect”:

This button is responsible for initiating a socket connection between a client machine and a host machine so the information about the processes is passed to the file on host machine from where the information can be taken to add to listbox.

1. **Processes**



This tab consists of 1 listbox which receives data from psutil library about the processes which are running on the machine in a hierarchical order of importance.

## Database

The database is created using same library as to create logs, however it does not format it. Whenever you launch an application it opens appDB.db file (if it does not exist, it creates it) in the same directory from where program was launched. It uses sqlite3 to create and handle databases. The database stores information about system state, similar to logs. It has 4 tables: CPU, RAM, diskio and networkio.

|  |  |  |  |
| --- | --- | --- | --- |
| CPU table | RAM table | Diskio table | Networkio table |
| Date | Date | Date | Date |
| CTX switches | Total RAM | Read count | Bytes sent |
| Interrupts | Available RAM | Write count | Bytes received |
| Software interrupts | Used RAM | Read bytes | Packets sent |
| System calls | Free RAM | Write bytes | Packets received |
| Usage percentage | Usage percentage | Read time | Errin (number of errors while sending) |
|  |  | Write time | Errout (number of errors while receiving) |
|  |  |  | Dropin (number of drops while sending) |
|  |  |  | Dropout (number of drops while receiving) |