Mutex: The System Metrics Analyzer

Synopsis Report
in Partial Fulfilment of the Requirements
for the Course of
Minor Project - II
In
Third year – Sixth Semester
of **Bachelor of Technology**
Computer Science & Engg.
Specialization in
DevOps and CCVT.

Under the guidance of

Ms. Avita Katal
Assistant Professor – Senior Scale
Department of Virtualization

By

500067035	**R171218058**	**Kshitiz Saini**
500068520	**R171218063**	**Muskaan Madan**
500067409	**R110218131**	**Saloni Saxena**
500068293	**R110218107**	**Pratyusha Agarwal**

UNIVERSITY WITH A PURPOSE

Department of Cybernetics and Virtualization
SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
Bidholi Campus, Energy Acres, Dehradun – 248007
January, 2021

School of Computer Science

University of Petroleum & Energy Studies, Dehradun

Project Proposal Approval Form (2020-2021)

Minor 2

Project Title:
Mutex: The System Metrics Analyzer

Abstract

System Metrics also referred as System Health allows the user to measure the system's vital resources as well as the system performance in real time. System Metrics plays an important role in the IT Industry as it helps the IT Support team to take the decisions based on system health and checks regularly if the devices are performing at the levels required of it and is it sufficient to satisfy the user and client demands. In the project ***"Mutex: The System Metrics Analyzer"***, we aim to build a Command-Line based system analytics tool to visualize the live health of our system.

Keywords: *System Health, System Analytics, Performance, IT Support, Real-Time, Metrics, Process, Thread, Memory, CPU*

TABLE OF CONTENT

1. INTRODUCTION

As an organization grows, workforce, resources, systems, services, and infrastructure also tend to grow considerably and it becomes difficult to maintain each of the system. Maintaining system health is a major concern for most of the IT Industries today as we need to ensure that the different system element services are running smoothly to keep the IT services running right. The primary reason is that while using any software, many users notice the performance problem as soon as it arises. They need to get it resolved quickly and find the cause for the issue. System Monitoring Software helps in resolving those issues, which may lead to a significant break in the system.

This project focuses on implementing a System Metrics Analyzer. The project mainly focuses on fetching data through system files, processing and dumping it highlighting the features of **Linux System Architecture.** The project will use the Object-Oriented Approach as well as handling and processing the data from the system files. The entire system analytics (processes and threads) will be reflected as the output of the project following the Command-Line approach.

The project focuses on the practical implementation of data handling and deployment of the project where the concepts of Agile as well as Waterfall Development will be used. Alongside, implementing the concept of Object-Orientation, UML and Command-Line will also be in consideration.

In Linux System architecture, everything is a file and all the files and directories appear under the root (/) directory, even if they are stored on different physical or virtual devices.

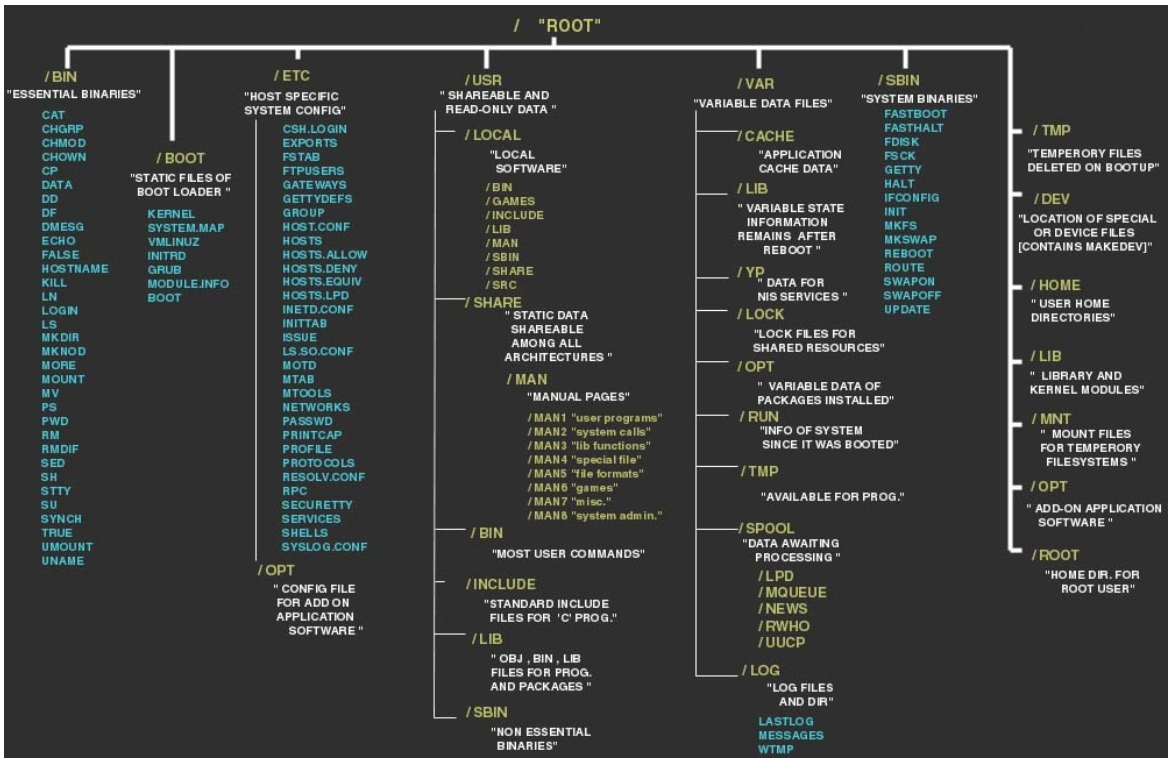


Figure (1) Linux Filesystem Architecture [6]

/proc is a virtual filesystem in Linux System architecture which stores the process as well as the kernel information arranged in form of files, which are automatically generated and populated by the Linux System on the fly.

- /proc contains the information about system process.
- It is a pseudo filesystem which contains the information about the running processes stored as, /proc/<PID>, where PID is the Process-ID of a particular process.
- There are files like /proc/uptime which contains the information about the system resources.

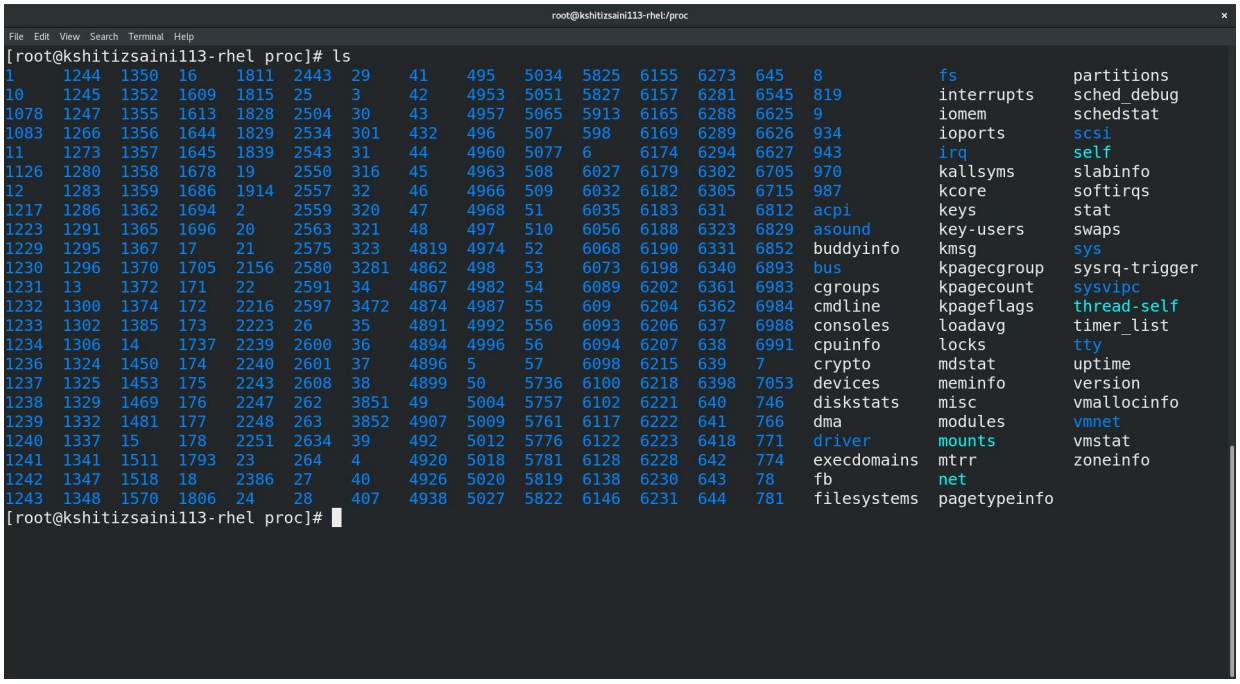


Figure (2) proc folder in Linux

2. LITERATURE REVIEW

1) In [1], the author Benjamin Maytal has talked about Real-Time manager and its importance. The author explained the concept of how a task service manages allocating time between real-time tasks and the other tasks, for selecting the service levels of real-time tasks in response to the activity levels of other tasks, and for invoking the real-time operating system to activate and control real-time tasks. The author also discussed about monitoring the CPU at a first time to determine the real-time tasks being performed and monitoring the performance counters which determine the activity level of the computer.

2) In [2], the authors Rolando Ingles, Piotr Perek, Mariusz Orlikowski and Andrej Apieralski explained multithreaded framework and its capability to process all the data produced by the source to ensure the highest level of accuracy, especially when it deals with hard real-time system monitoring. In this paper the authors have presented the comparison between various C++ frameworks that using multithreading technology and ring-buffer data structure allow data transfer in concurrent way. The comparison is based on the time interval between the instant when data is published and the instant when the data is gathered.

3) In [3], the authors Konstantin S. Stefanov , Alexey A. Gradskov have presented analysis of CPU usage data properties and their possible impact on performance monitoring. In this paper the authors have analyzed data provided by the Linux kernel and how CPU load level is calculated based on these data. This paper explains how CPU usage data is obtained and experiments such as measuring intervals between CPU usage data changes, etc.

4) In [4], the author Andrew Bishop has explained in detail about the /proc file system—what it is, and how it can be used. There is also a description of the program ProcMeter that uses the /proc file system to display useful information.

3. PROBLEM STATEMENT

It is a challenge for IT Organizations to be proactive all the time and to maintain the Infrastructure of their organization. Even today, IT organization spend more time reacting to the problems rather than identifying them before causing the disruption. For IT teams to run at optimal performance and to prevent errors, proactive IT infrastructure monitoring is very important to offer top-notch services to the clients and the users. To tackle the issue of monitoring, we are designing an application which can show the statistics and health of our IT Infrastructure.

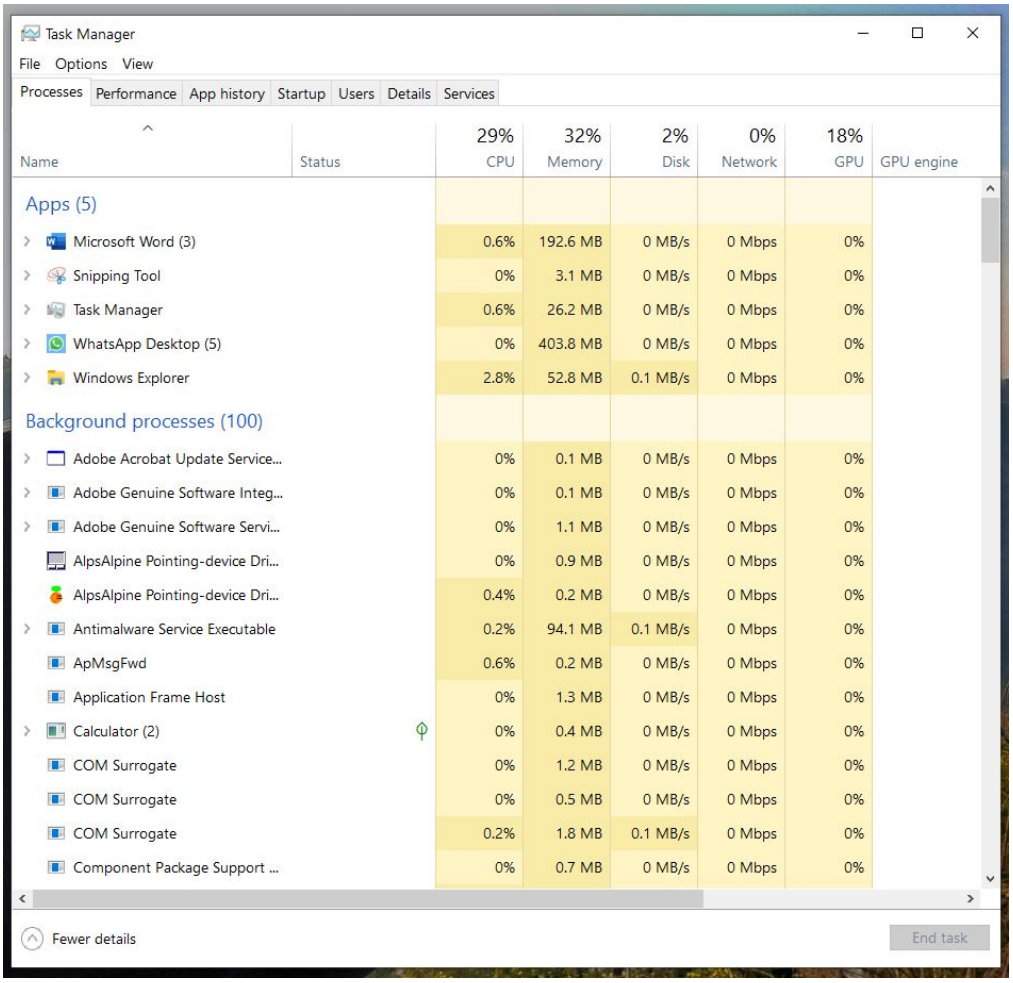


Figure (3) Task Manager (Windows System Metrics Analyzer)

4. OBJECTIVES

The main objective of the project is to implement the System Metrics Analyzer.

Sub-objectives for the project are:

- ➔ Fetching the data from system files.
- ➔ Processing the data to gather useful information.
- ➔ Implementing the System Metrics Analyzer as a Command Line utility.
- ➔ Packaging the whole project and deploying it using a CI-CD pileline.

5. METHODOLOGY

This project is blending the Agile Methodology and Waterfall Methodology of software development as it's rare to find all the qualities in a single software development methodology. There are different ways to implement a waterfall methodology, including iterative waterfall, which still practices the phased approach but delivers in smaller release cycles. The project used the Agile methodology for the Development in Build part of the project to take the advantage of the Documentation part of the Waterfall methodology as well as utilizing the sprints as a part of Agile workflow.

Overall, the time for the project is dedicated to an approach where the beginning time is dedicated towards the requirement analysis and the documentation part and during the implementation part all the team members are following their dedicated sprints cycles to implement the functionality. After the implementation, testing is to be done for the whole application. Finally, the application is deployed with the documentation.

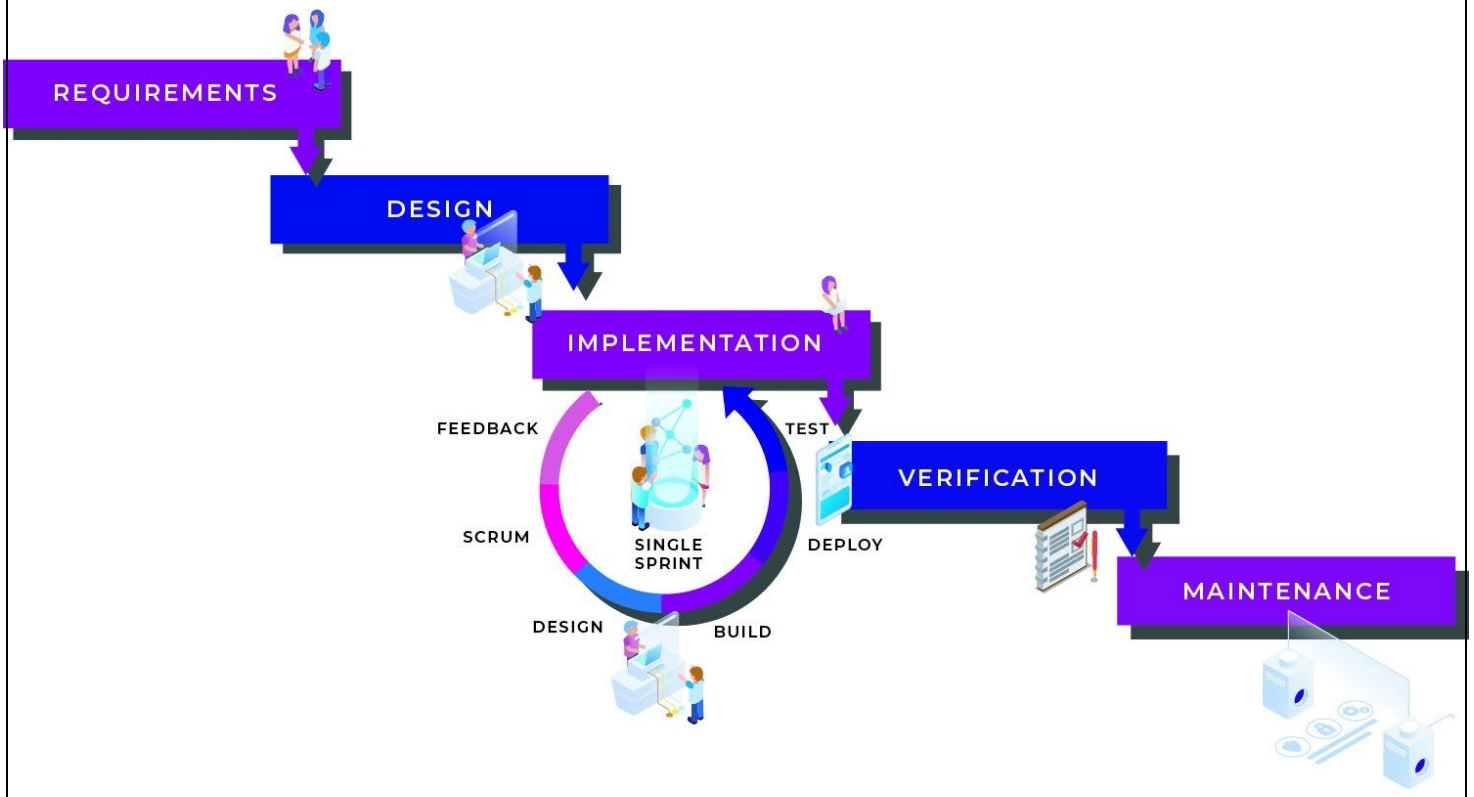


Figure (4) Hybrid Waterfall-Agile Model [5]

Further, we would like to package our project in a Full-Fledged DevOps Pipeline using the DevOps tools like, Jenkins, Ansible and AWS Cloud.

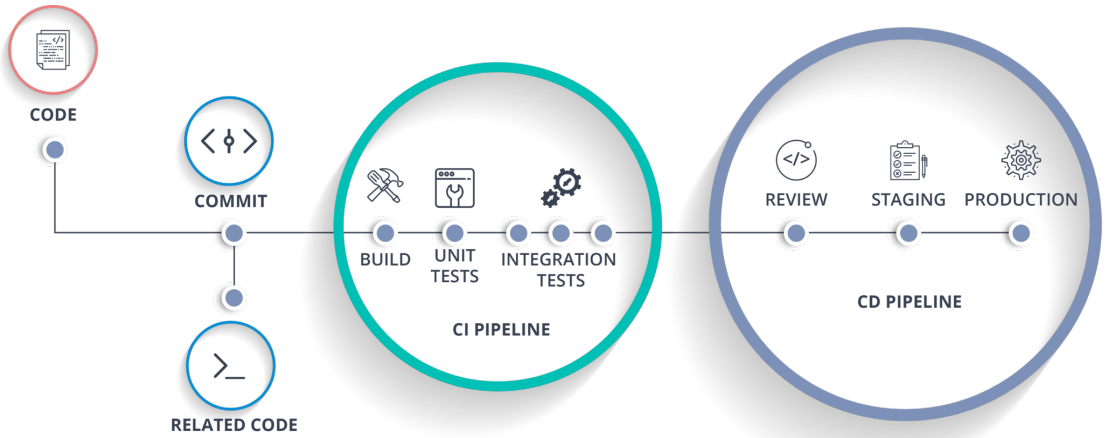


Figure (5) CI-CD Pipeline

We will maintain the project on GitHub and will fetch it using Jenkins at regular intervals to create it's build and further we will use Jenkins pipeline to deploy the Docker Image on Docker Hub. For the delivery pipeline, we will use Ansible Configuration Management tool to automate its deployment on various instances.

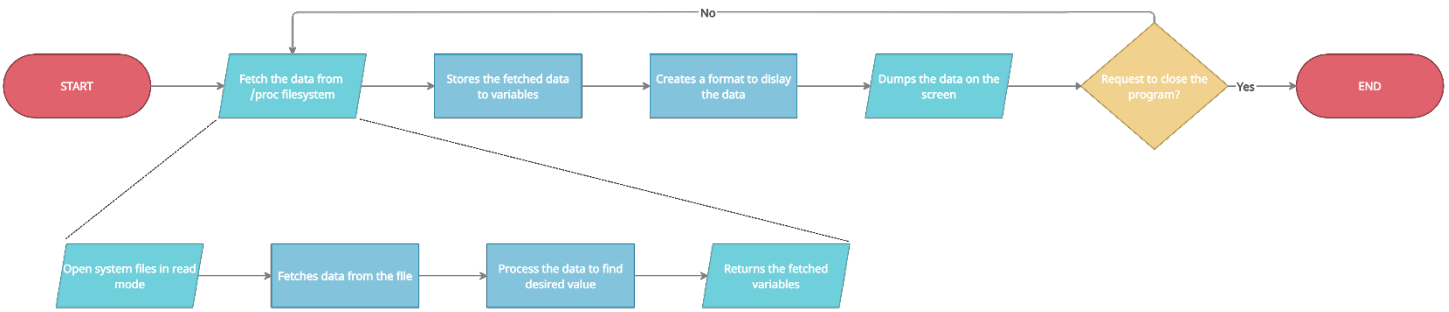


Figure (6) Flowchart

First of all, this System Metrics Analyser process and filter data from the desired files such as /proc/uptime, /proc/<PID>, etc stored in /proc directory. After processing the data, it gets stored in some variables and then operations are performed on the fetched data. Now, this data is ready to get displayed. ncurses display library creates a format to display the data on the screen and the data is dumped on the screen.

Here, the process of fetching the data from the /proc filesystem is further divided into 4 subprocesses, where first of all, the file is opened in read mode, further the data is read from the files and processed to fetch the desired values from the data. After fetching all the desired values, the data is returned for further processing.

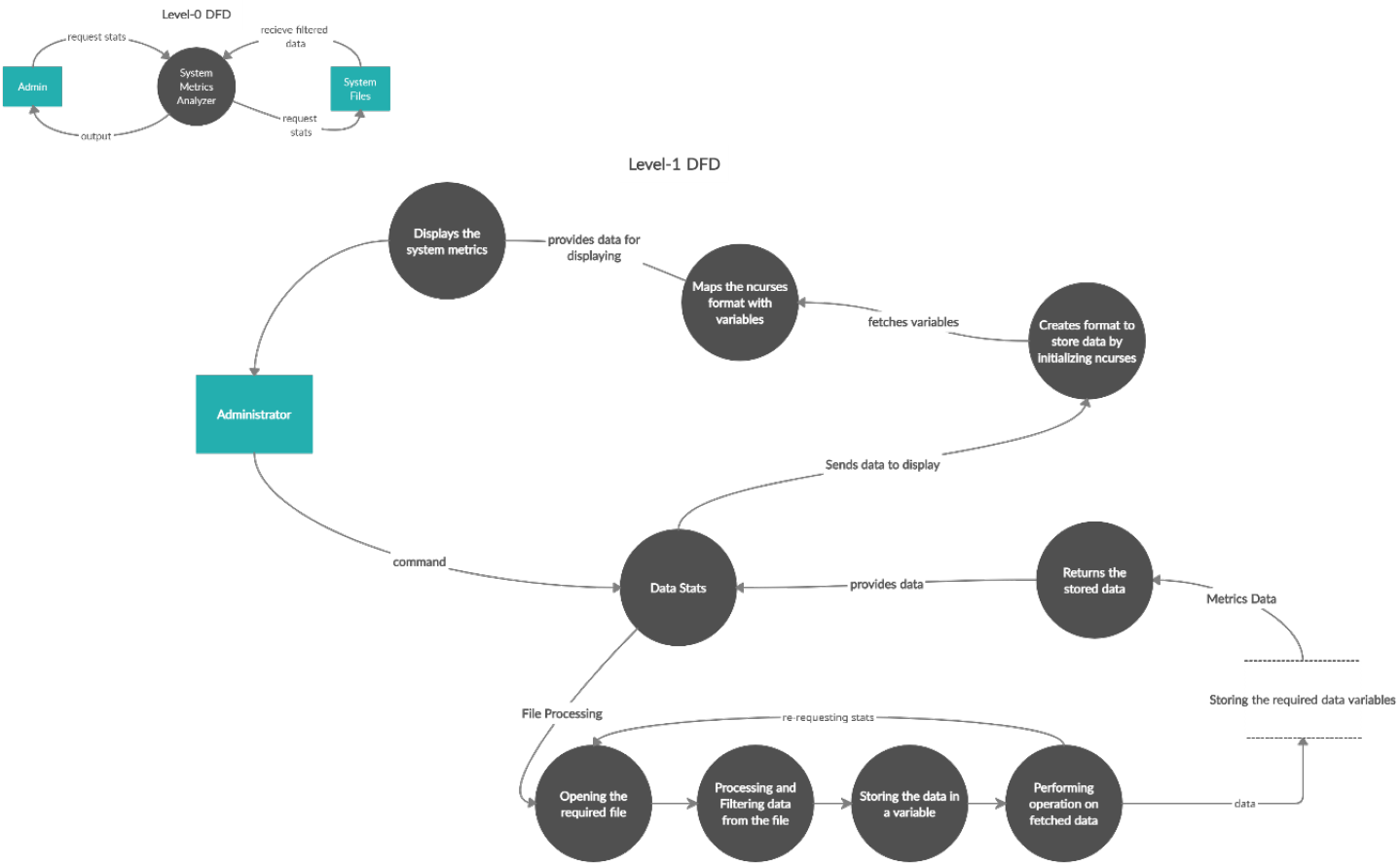


Figure (7) Data Flow Diagram

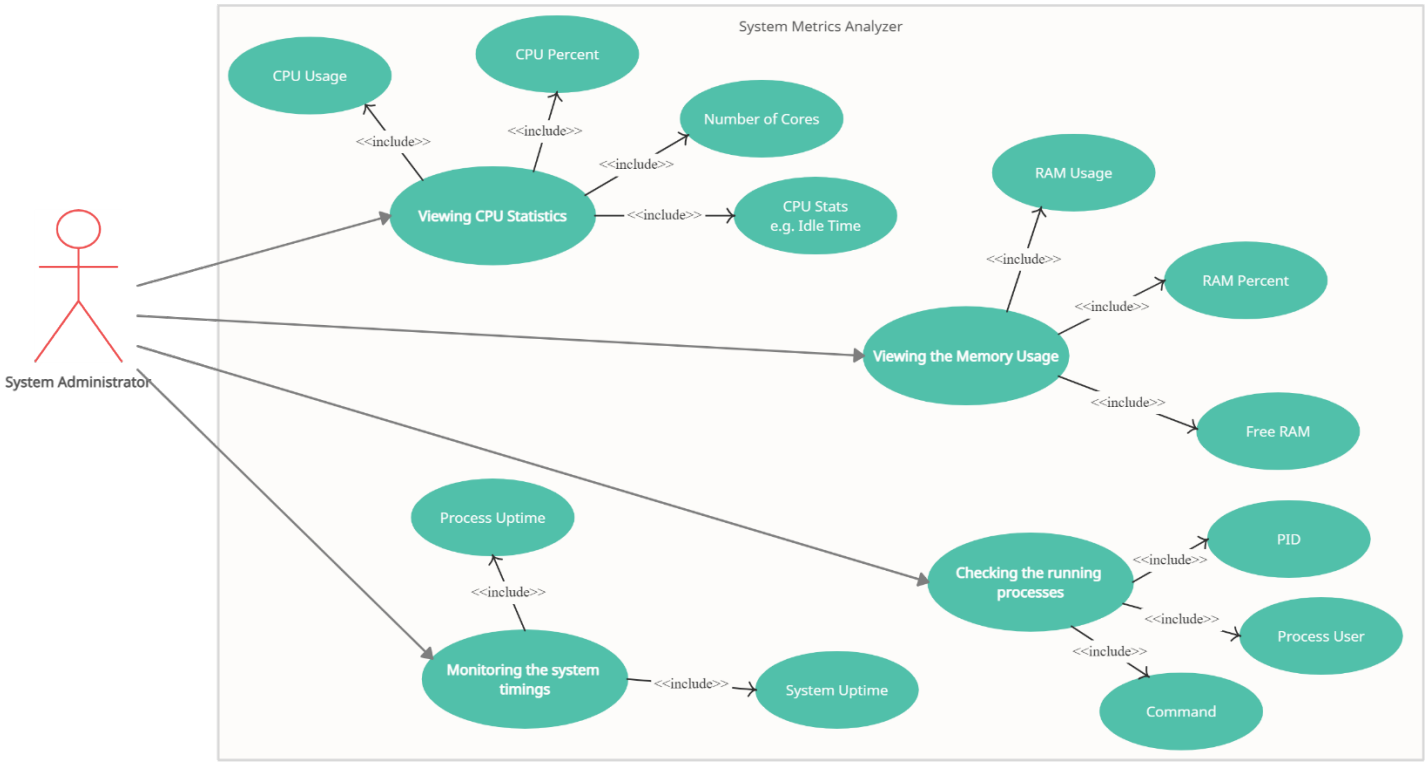


Figure (8) Use Case Diagram

6. SYSTEM REQUIREMENTS

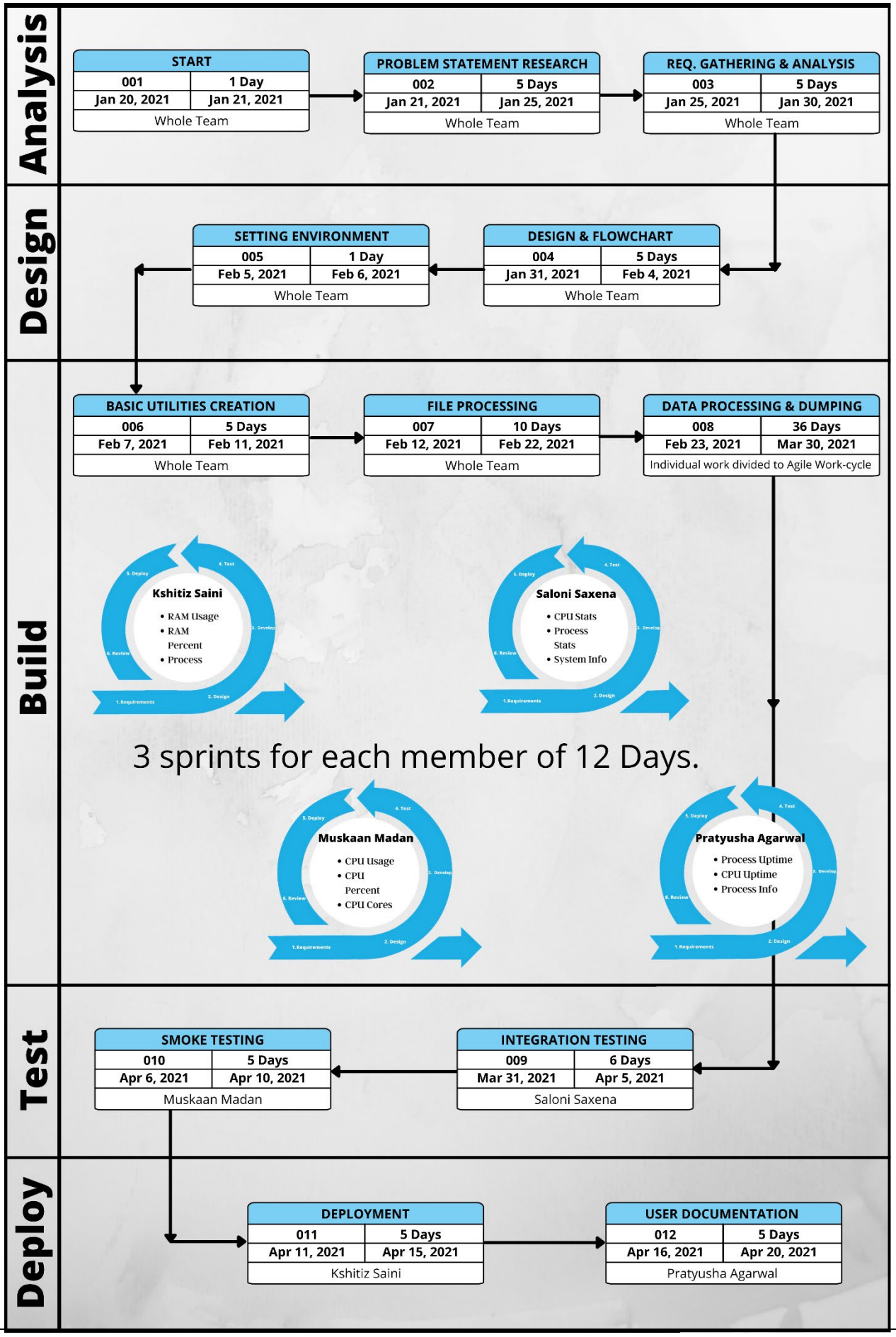
1. Software

- g++ Compiler
- Any flavor of Linux

2. Hardware

- 512 MB RAM
- i3 5th Generation or above processor

7. PERT CHART


Analysis
START
001
1 Day
Jan 20, 2021
Jan 21, 2021
Whole Team
PROBLEM STATEMENT RESEARCH
002
5 Days
Jan 21, 2021
Jan 25, 2021
Whole Team
REQ. GATHERING & ANALYSIS
003
5 Days
Jan 25, 2021
Jan 30, 2021
Whole Team
Design
SETTING ENVIRONMENT
005
1 Day
Feb 5, 2021
Feb 6, 2021
Whole Team
DESIGN & FLOWCHART
004
5 Days
Jan 31, 2021
Feb 4, 2021
Whole Team
Build
BASIC UTILITIES CREATION
006
5 Days
Feb 7, 2021
Feb 11, 2021
Whole Team
FILE PROCESSING
007
10 Days
Feb 12, 2021
Feb 22, 2021
Whole Team
DATA PROCESSING & DUMPING
008
36 Days
Feb 23, 2021
Mar 30, 2021
Individual work divided to Agile Work-cycle
Kshitiz Saini
• RAM Usage
• RAM Percent
• Process
Saloni Saxena
• CPU Stats
• Process Stats
• System Info
3 sprints for each member of 12 Days.
Muskaan Madan
• CPU Usage
• CPU Percent
• CPU Cores
Pratyusha Agarwal
• Process Uptime
• CPU Uptime
• Process Info
Test
SMOKE TESTING
010
5 Days
Apr 6, 2021
Apr 10, 2021
Muskaan Madan
INTEGRATION TESTING
009
6 Days
Mar 31, 2021
Apr 5, 2021
Saloni Saxena
Deploy
DEPLOYMENT
011
5 Days
Apr 11, 2021
Apr 15, 2021
Kshitiz Saini
USER DOCUMENTATION
012
5 Days
Apr 16, 2021
Apr 20, 2021
Pratyusha Agarwal

8. REFERENCES

[1] B. Maytal, M. Zion, Israel "Real-time task manager for a personal computer," United States Patent 6,092,095, Jul.18, 2000

[2] R. Ingles, P.Perek, M. Orlikowski and A. Napieralski, "A simple multithreaded C++ framework for high-performance data acquisition systems," 2015 22nd International Conference Mixed Design of Integrated Circuits & Systems (MIXDES), Torun, 2015, pp. 153-157, doi: 10.1109/MIXDES.2015.7208501

[3] Konstantin S. Stefanov, Alexey A. Gradskov "Analysis of CPU Usage Data Properties and their possible impact on Performance Monitoring," SuperFri.org, doi: 10.14529/jsfi160405.

[4] Andrew M. Bishop, "The /proc File System And ProcMeter," Linux Journal, April 1,1997

[5] Benjamin ZY Tan, "Effective Agile + Waterfall Hybrid Project Management", medium.com

[6] BlackMoreOps, "Linux file system hierarchy v1.0", https://www.blackmoreops.com/2015/02/14/linux-file-system-hierarchy/

[7] SolidStudio, "Benefits of CI/CD Pipelines," https://solidstudio.io/blog/ci-cd-pipelines.html

Synopsis Draft verified by

Ms. Avita Katal

Assistant Professor-Senior Scale
Department of Virtualization
School of Computer Science

Dr. Monit Kapoor

Head of Department
Department of Cybernetics
School of Computer Science

Project Guide
(Ms. Avita Katal)

HOD
(Dr. Monit Kapoor)