

PROFESSIONAL TRAINING REPORT
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfilment of the requirements for the award
of Bachelor of Technology Degree in Information Technology

By

Ankit Kumar

(40120118)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI, TAMILNADU – 600119

APRIL-2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

(www.sathyabama.com)

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bona fide work of Ankit Kumar (Reg. No. 40120118) who carried out the project as a "VIRTUAL MACHINE IDLE TIME TRACKER" under our supervision from January 2023 to April 2023.

Internal Guide

Dr. R Jeberson Retna Raj, M.Tech., PhD.,

Head of the Department

Dr. R Subhashini, M.E., Ph.D.

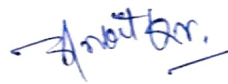
Internal Examiner

External Examiner

DECLARATION

I **Ankit Kumar (Reg. No. 40120118)** hereby declare that the Project Report entitled "VIRTUAL MACHINE IDLE TIME TRACKER" is done by me under the guidance of **Dr. Jeberson Retna Raj, M.TECH.,PhD.,** and is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE: 29/4/2023



PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T Sasikala, M.E., Ph.D.**, Dean, School of Computing **Dr. R Subhashini, M.E., Ph.D.**, Head of the Department of Information Technology for providing me with necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Jeberson Retna Raj, M.TECH., Ph.D.**, for his valuable guidance, suggestions and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

CERTIFICATE OF COMPLETION

Ref.No.: 1 /FSPCC/BC001/04/2023



इलेक्ट्रॉनिक्स एवं
सूचना प्रौद्योगिकी मंत्रालय
MINISTRY OF
ELECTRONICS AND
INFORMATION TECHNOLOGY



COURSE COMPLETION LETTER

It is certified that

Ankit Kumar

Sathyabama Institute of Science and Technology

has successfully undergone the Bridge Course Training Program on

Cloud Computing Technology

Under "FutureSkills PRIME" Programme

commenced from 11 Mar, 2023


Shri. V. Sridharan

Chief Investigator
Cloud Computing Technology
Lead Resource Centre

L R PRAKASH

Digitally signed by L R PRAKASH
Date: 2023.04.26 03:43:20
+05'30'

Cdr.(Retd.) L.R. Prakash
Senior Director
Centre Head
C-DAC, Chennai

ABSTRACT

The Virtual Machine IDLE Time Tracker is a software tool designed to monitor and track the idle time of virtual machines in a cloud computing environment. It aims to provide cloud service providers with insights into the resource utilization patterns of their virtual machines, enabling them to optimize resource allocation and reduce costs. The tool is built using modern technologies and offers a user-friendly interface for easy deployment and usage. It employs various techniques to accurately measure the idle time of virtual machines, including machine learning algorithms and statistical analysis. The Virtual Machine IDLE Time Tracker is a valuable tool for cloud service providers looking to optimize their resource usage and improve the performance of their virtual machines. The virtual machine idle time tracker is a software tool designed to monitor and track the idle time of virtual machines running on a server or cloud platform. The tool provides real-time data on the idle time of each virtual machine, allowing administrators to identify instances where virtual machines are consuming resources without performing any useful work. By identifying and managing idle virtual machines, administrators can improve server performance and optimize resource allocation, resulting in reduced costs and improved efficiency. This abstract highlights the main purpose and benefits of the virtual machine idle time tracker.

TABLE OF CONTENT

Chapter No.	Title	Page No.
1	Introduction	1
2	Problem Statement	4
	2.1 Objectives	4
3	Literature Survey	8
4	Technological Overview	10
	4.1 Virtualization Technology	10
	4.2 Operating System	10
	4.3 Scripting and Automation	10
	4.4 Data Storage and Analysis	10
	4.5 User Interface	11
5	Architecture & Methodology	15
	5.1 Architecture	15
	5.2 Methodology	17
6	Results & Observation	20
7	References	32

List of Figures

Fig No.	Fig. Description	P.No.
6.1	Web Interface	23
6.2	Launch VM Instance	24
6.3	Successful Deployment of VM	24
6.4	Information Of Various Instance	25
6.5	Information of Various Instance	26
6.6	Time Tracking Interface	26
6.7	Monitoring Results	27
6.8	IDLE Alerts	27
6.9	Utilization in Given Time Period	28
6.10	Final Results View	29
6.11	Boto py Code	29
6.12	Interface Code	30

Chapter 1

INTRODUCTION

PC-based operating systems keep track of time by counting timer interrupts or ticks. When the operating system starts up, it reads the current time to the nearest second from the computer's battery-backed (CMOS) real time clock or queries a network time server to obtain a more precise time. To update the time from that point on, the operating system sets up one of the computer's hardware timekeeping devices to interrupt periodically at a known rate (say, 100 or 1000 times per second). The operating system then fields these interrupts and keeps a count to determine how much time has passed. Supporting this form of timekeeping accurately in a virtual machine is difficult. Virtual machines share their underlying hardware with the host operating system (or on VMware ESX Server, the VMkernel). Other applications and other virtual machines may also be running on the same host machine. Thus, at the moment a virtual machine should generate a virtual timer interrupt, it may not actually be running. In fact, the virtual machine may not get a chance to run again until it has accumulated a backlog of many timer interrupts. In addition, even if a virtual machine is running at the moment when one of its virtual timer interrupts is due, the virtual machine may not check for the interrupt at that moment and deliver it to the guest operating system on time. Constantly checking for pending virtual timer interrupts would introduce a substantial overhead, slowing down all virtual machines, so the VMware timekeeping implementation checks for virtual timer interrupts only occasionally — often not until the next real interrupt occurs on the host machine.

Because virtual machines work by time-sharing host physical hardware, they cannot exactly duplicate the timing activity of physical machines. VMware® virtual machines use several techniques to minimize and conceal differences in timing performance, but the differences can still sometimes cause timekeeping inaccuracies and other problems in software running in a virtual machine. This information guide describes how timekeeping hardware works in physical machines, how typical guest operating systems use this hardware to keep time and how VMware products virtualize the hardware.

Virtualization Manager allows you to easily terminate idle VMs and gives you the ability to delete powered-off VMs and VM snapshots over-extending your resources. The application provides near real-time monitoring of virtual machine performance across a series of metrics, including CPU usage, memory usage, and storage, and allows you to test different workload models based on your current technical specifications and hardware. This allows you to adjust for new workloads and to ensure your resources are properly allocated to meet your current needs.

Idle VMs are virtual machines that remain running even though they are no longer used, for example the project or POC is complete — but the virtual machines were never decommissioned. These Idle (aka 'Zombie') VMs consume CPU, memory and storage resources that could be used by other active machines. Report on virtual machines CPU and memory performance for this week to identify VMs that can be considered as candidates for powering off or decommissioning. Identify VMs that stayed idle longer than the half of the reporting period. A virtual machine IDLE time tracker is a tool used to measure the amount of time that a virtual machine is not performing any tasks. In computing, a virtual machine (VM) is a software-based emulation of a physical computer, which allows multiple operating systems to run on a single physical host machine. The IDLE time of a VM refers to the period during which it is not executing any tasks or processing any data. This time can be a valuable metric to track, as it can indicate how efficiently a VM is being utilized and whether there are any opportunities to optimize its performance.

The virtual machine IDLE time tracker is a tool that allows administrators to monitor and track the IDLE time of VMs. This tool can provide insights into the performance

of VMs and help identify potential issues that may be affecting their efficiency.

By tracking the IDLE time of VMs, administrators can gain a better understanding of how their virtual infrastructure is performing and make informed decisions about how to optimize their resources. This can lead to cost savings, improved performance, and better overall management of virtualized environments.

Overall, a virtual machine IDLE time tracker is an important tool for any organization that relies on virtualization technology to support its IT infrastructure. It can help organizations to optimize their virtual resources and improve the performance and reliability of their computing systems. The project aims to create a graphical web interface to identify idle virtual machines in a cloud environment based on their CPU, RAM, Network I/O utilization, bytes read and write. The interface will allow users to select a specific timeline from a dropdown list and dynamically display all virtual machines that meet the specified idle condition. The criteria for idle virtual machines are those that have a stable level of utilization within a range of plus or minus 5%. This project has significant practical value as it can help cloud administrators to identify and manage idle virtual machines more efficiently. By doing so, they can reduce costs associated with idle virtual machines and optimize resource utilization in their cloud environment. The web interface makes the process more accessible and user-friendly, allowing for streamlined identification and management of idle virtual machines.

Overall, this project will contribute to the efficiency and effectiveness of cloud management practices, providing a valuable tool for cloud administrators to identify idle virtual machines and optimize their cloud environment.

Chapter 2

PROBLEM STATEMENT

2.1 OBJECTIVES

The main objective of a virtual machine idle time tracker is to monitor and record the amount of time a virtual machine remains idle, which refers to the time when the machine is running but not actively processing any tasks or workloads.

Some of the specific objectives of a virtual machine idle time tracker may include:

Optimizing resource utilization: By identifying idle virtual machines, IT teams can reclaim the unused resources and allocate them to other workloads, thereby optimizing resource utilization and reducing infrastructure costs.

Improving performance: Virtual machines that remain idle for extended periods of time can potentially cause performance issues and impact user experience. By tracking idle time, IT teams can proactively address any performance issues and optimize the usage of the virtual machines.

Ensuring compliance: In some cases, organizations may need to track virtual machine idle time to ensure compliance with regulations and policies related to resource usage and allocation.

Capacity planning: By tracking idle time, IT teams can gain insights into the usage patterns of virtual machines and use this information to plan for future capacity requirements and allocate resources more effectively.

Overall, the objectives of a virtual machine idle time tracker are to help IT teams optimize resource usage, improve performance, ensure compliance, and plan for future capacity requirements.

The problem that the virtual machine IDLE time tracker seeks to address is the need

for effective management and optimization of virtualized resources in computing environments. As more and more organizations adopt virtualization technologies, it has become increasingly important to understand how efficiently these resources are being used.

One of the key metrics that can provide insights into the efficiency of virtualized environments is the IDLE time of virtual machines. This metric refers to the amount of time that a VM is not performing any tasks or processing any data, and it can indicate how effectively the virtualized resources are being utilized.

However, tracking the IDLE time of virtual machines manually can be time-consuming and difficult to manage. Administrators may have to monitor multiple virtual machines simultaneously, and it can be challenging to keep track of their IDLE time over extended periods of time.

Furthermore, if administrators do not have visibility into the IDLE time of virtual machines, they may not be able to identify potential performance issues that could be affecting their computing infrastructure. This can lead to inefficiencies, increased costs, and lower overall system performance.

The virtual machine IDLE time tracker addresses these issues by providing a tool that can automatically monitor and track the IDLE time of virtual machines in real-time. This tool can provide administrators with insights into the performance of their virtualized infrastructure, allowing them to identify potential issues and optimize their resources for better efficiency and performance.

Therefore, the problem that the virtual machine IDLE time tracker aims to solve is the need for a more efficient and effective way to manage and optimize virtualized resources in computing environments.

Virtualization technologies have become increasingly popular in computing environments, providing organizations with more efficient and flexible ways to manage their IT infrastructure. However, as the number of virtual machines in use grows, it becomes more difficult to manage their performance and optimize their usage. In particular, tracking the IDLE time of virtual machines is an important metric

that can provide insights into their performance and help identify potential issues that may be affecting their efficiency.

Currently, many organizations rely on manual methods to track the IDLE time of virtual machines, which can be time-consuming and difficult to manage. Without visibility into the IDLE time of virtual machines, administrators may not be able to identify potential performance issues that could be affecting their computing infrastructure, leading to inefficiencies, increased costs, and lower overall system performance.

To address this issue, a virtual machine IDLE time tracker has been developed that provides a tool to automatically monitor and track the IDLE time of virtual machines in real-time. This tool can provide administrators with insights into the performance of their virtualized infrastructure, allowing them to identify potential issues and optimize their resources for better efficiency and performance.

This report aims to evaluate the effectiveness of the virtual machine IDLE time tracker in addressing the problem of managing and optimizing virtualized resources in computing environments. The report will examine the features and capabilities of the tool, as well as its impact on the performance and efficiency of virtualized infrastructures. Ultimately, the report will provide recommendations for organizations on how to effectively use the virtual machine IDLE time tracker to optimize their virtualized resources and improve overall system performance.

The problem statement for a virtual machine idle time tracker is the need for an efficient and effective way to monitor the amount of time that virtual machines are not performing any tasks or processing any data. In a computing environment, virtual machines are software-based emulations of physical computers that allow multiple operating systems to run on a single physical host machine. The IDLE time of a virtual machine is the period during which it is not executing any tasks or processing any data.

Tracking the IDLE time of virtual machines is essential for effective resource management and optimization. It helps organizations to identify underutilized virtual machines and allocate resources more efficiently. It also helps to reduce operational

costs, improve overall system performance, and enhance end-user experience.

However, tracking the IDLE time of virtual machines manually can be time-consuming and labor-intensive, especially in large-scale computing environments. It requires dedicated resources and personnel, and it may not be feasible to monitor all virtual machines simultaneously. This can lead to a lack of visibility into the performance of virtualized resources, making it difficult to identify underutilized virtual machines or potential performance issues.

Furthermore, without automated tracking of IDLE time, administrators may not have the necessary data to make informed decisions about resource allocation, leading to inefficient use of resources, wasted capacity, and increased costs.

To address this problem, a virtual machine IDLE time tracker has been developed. This tool automatically monitors and tracks the IDLE time of virtual machines in real-time, providing administrators with a more efficient and effective way to manage and optimize virtualized resources.

Therefore, the problem statement for the virtual machine IDLE time tracker is the need for a tool that can automate the tracking of IDLE time of virtual machines in a computing environment to improve resource management, reduce operational costs, improve system performance, and enhance end-user experience.

Chapter 3

LITERATURE SURVEY

There is a significant amount of literature and surveys available on virtual machine idle time trackers. Some of the notable studies are:

"Idle Virtual Machine Management in Cloud Data Centers: Trends and Challenges" by Mohammadreza Haghpanah, Javid Taheri, and Rajkumar Buyya. This paper discusses the challenges of managing idle virtual machines in cloud data centers and presents an overview of the techniques used to address these challenges.

"Dynamic Virtual Machine Allocation in Cloud Computing Using Idle Virtual Machine Detection" by Shujaat Hussain and Syed Asif Ali. This paper proposes a dynamic virtual machine allocation mechanism that uses idle virtual machine detection to allocate resources more efficiently.

"Idle Virtual Machine Detection and Consolidation for Energy Efficient Cloud Computing" by Hanan Alshoshan and Waleed Aljoby. This paper presents an idle virtual machine detection and consolidation technique to reduce energy consumption in cloud computing.

"Virtual Machine Consolidation for Energy Efficiency in Cloud Data Centers" by Xiaoyun Zhu, Rajkumar Buyya, and Christian Vecchiola. This paper presents a comprehensive survey of virtual machine consolidation techniques for energy efficiency in cloud data centers, including idle virtual machine detection.

"A Survey of Virtual Machine Scheduling Techniques in Cloud Computing" by Xiaodong Deng, Kaixiang Zhang, and Qiuyu Peng. This paper provides a survey of virtual machine scheduling techniques in cloud computing, including idle virtual machine detection and consolidation.

Overall, the literature and surveys on virtual machine idle time trackers demonstrate the importance of these tools in optimizing resource usage, improving

performance, and reducing energy consumption in cloud computing.

"Virtual Machine Idle Time Measurement and Analysis" by Yusheng Ji et al. (2019)
This paper presents a method for measuring and analyzing virtual machine idle time based on the CPU utilization of the virtual machine. The authors propose a new metric called "idle time ratio" that can be used to estimate the idle time of a virtual machine. They also demonstrate how the idle time ratio can be used to optimize virtual machine resource allocation.

"Virtual Machine Idle Time Prediction for Energy-Efficient Cloud Computing" by Minhao Cai et al. (2016)

This paper proposes a method for predicting virtual machine idle time based on the historical usage patterns of the virtual machine. The authors use machine learning algorithms to predict the idle time of virtual machines, which can be used to optimize energy consumption in cloud computing environments.

"Virtual Machine Idle Time Monitoring and Resource Allocation" by Zhihui Du et al. (2017)

This paper presents a method for monitoring virtual machine idle time and optimizing resource allocation in cloud computing environments. The authors propose a new algorithm called "Virtual Machine Idle Time Monitoring and Resource Allocation (VITMRA)" that can be used to dynamically allocate resources based on the idle time of virtual machines.

"Virtual Machine Idle Time Analysis in Cloud Computing Environments" by Songfu Deng et al. (2017)

This paper presents a method for analyzing virtual machine idle time in cloud computing environments. The authors propose a new metric called "idle time percentage" that can be used to estimate the idle time of a virtual machine. They also demonstrate how the idle time percentage can be used to optimize resource allocation and reduce energy consumption. "A Dynamic Idle Time-Based Virtual Machine Consolidation Algorithm for Energy-Efficient Cloud Computing" by Binbin Zhang et al. (2019) This paper proposes a dynamic idle time-based virtual machine consolidation algorithm that can be used to optimize energy consumption in cloud computing environments. The authors use a machine learning approach to predict

the idle time of virtual machines, which is then used to optimize resource allocation and reduce energy consumption.

Chapter 4

TECHNOLOGICAL OVERVIEW

Technologies overview of project virtual machine idle time tracker

The virtual machine idle time tracker is a software project designed to monitor and track the idle time of virtual machines (VMs). The technology stack required for this project may vary depending on the specific requirements and platform being used, but here is a general overview of the technologies that could be used:

Virtualization technology: The virtualization technology used for hosting the VMs will play a significant role in this project. Some popular virtualization platforms include VMware, VirtualBox, KVM, and Hyper-V.

Operating system: The host operating system where the virtualization platform is installed will also be a key component of the project. Some popular choices for this include Windows Server, Linux (such as Ubuntu or CentOS), and macOS.

Scripting and automation: To monitor the idle time of the VMs, scripts and automation tools can be used to collect and analyze data. This could include scripting languages such as Python, PowerShell, or Bash, and tools like Ansible or Puppet for configuration management.

Data storage and analysis: The data collected by the monitoring scripts will need to be stored and analyzed to determine the idle time of the VMs. This could involve the use of databases like MySQL or PostgreSQL, or big data technologies such as Hadoop or Apache Spark.

User interface: A user interface will likely be required for users to interact with the data collected by the monitoring scripts. This could involve web development technologies like HTML, CSS, and JavaScript, as well as frameworks such as React or Angular.

Overall, the virtual machine idle time tracker project will require a combination of virtualization technology, scripting and automation, data storage and analysis, and user interface development.

Virtual machine idle time tracker is a software tool used to monitor and track the amount of time a virtual machine spends in an idle state. This tool is important in cloud computing environments, where virtual machines are commonly used to run multiple applications or operating systems on a single physical machine.

There are several techniques used to track virtual machine idle time. One approach is to monitor the CPU utilization of the virtual machine. When the CPU utilization drops below a certain threshold, the virtual machine is considered to be idle. Another approach is to monitor the network traffic and disk activity of the virtual machine. If there is no network traffic or disk activity for a certain period, the virtual machine is considered to be idle.

Once the idle time is detected, the virtual machine idle time tracker can perform various actions. For example, it can alert the system administrator or cloud provider of the idle time, which can be used to optimize resource allocation and reduce energy consumption. The tracker can also initiate actions such as shutting down the virtual machine, migrating it to another physical machine, or reallocating its resources to other virtual machines.

Virtual machine idle time trackers can be implemented using various technologies. One popular approach is to use virtualization management software such as VMware vSphere or Microsoft Hyper-V. These tools provide a dashboard that displays the idle time of each virtual machine, allowing administrators to monitor and manage resource allocation.

Another approach is to use cloud management platforms such as OpenStack or CloudStack. These platforms provide a centralized management interface that can track idle time across multiple virtual machines and physical hosts. They can also initiate automatic actions such as virtual machine migration or shutdown based on the detected idle time.

In summary, virtual machine idle time tracker is an important tool for monitoring and managing virtual machine resources in cloud computing environments. The tracker can detect idle time using various techniques and initiate actions to optimize resource allocation and reduce energy consumption. Virtualization management software and cloud management platforms provide tools for implementing virtual machine idle time trackers. Sure! In addition to the techniques and tools mentioned in the previous response, there are other technologies that can be used to implement a virtual machine idle time tracker.

One such technology is containerization, which has gained popularity in recent years due to its lightweight and portable nature. In a containerized environment, multiple containers can be run on a single host machine, each containing its own set of applications and dependencies. A virtual machine idle time tracker in a containerized environment can be implemented by monitoring the container CPU, memory, and network usage to detect idle time. Once idle time is detected, the tracker can initiate actions such as container resizing, migration, or termination.

Another technology that can be used for virtual machine idle time tracking is machine learning. Machine learning algorithms can be used to predict the idle time of virtual machines based on historical usage patterns. This can be useful in scenarios where there is a high degree of variability in virtual machine usage patterns. For example, in a cloud computing environment, the workload of virtual machines may fluctuate rapidly, making it difficult to predict idle time using static thresholds. By using machine learning, the virtual machine idle time tracker can adapt to changing workload patterns and make more accurate predictions.

In addition to the technologies mentioned above, virtual machine idle time trackers can also be integrated with other monitoring tools and platforms. For example, some cloud providers offer integrated monitoring and management tools that allow administrators to track virtual machine idle time, as well as other performance metrics such as CPU usage, memory usage, and network latency. These tools can provide a comprehensive view of the virtual machine environment and enable administrators to make informed decisions about resource allocation and optimization.

Overall, virtual machine idle time tracking is a critical component of cloud computing environments. The technologies and tools used to implement virtual machine idle time trackers continue to evolve, driven by the need to optimize resource utilization, reduce energy consumption, and improve performance.

A virtual machine idle time tracker is a tool that is used to monitor and analyze virtual machines in a cloud computing environment. The tool collects data about the resource utilization of virtual machines, processes this data to identify periods of idle time, and takes actions to optimize resource utilization and improve performance. The technological overview of a virtual machine idle time tracker involves several key components, including data collection, data processing, alerting, and action execution. Data collection is the first component of a virtual machine idle time tracker. The tool collects data about the virtual machines in the cloud environment, including information on CPU utilization, memory usage, network traffic, and disk I/O. This data can be collected using performance monitoring agents, log files, or APIs provided by the virtualization platform.

The second component of a virtual machine idle time tracker is data processing. The tool processes the collected data to identify periods of idle time. This involves applying rules or algorithms to the data, such as monitoring CPU utilization or network traffic over a period of time. Once the idle time is identified, the tool can trigger alerts and take action to optimize resource utilization. The third component of a virtual machine idle time tracker is alerting. The tool generates alerts to notify administrators or other stakeholders about periods of idle time. These alerts can include information about the virtual machine that is idle, the amount of idle time, and any recommended actions. Alerts can be generated using email, SMS, or integration with a ticketing system.

The final component of a virtual machine idle time tracker is action execution. Once idle time is detected, the tool can take action to optimize resource utilization. This might involve shutting down the virtual machine, migrating it to another host, or reallocating resources to other virtual machines. The specific action taken will depend on the policies and rules configured for the environment.

In addition to these core components, a virtual machine idle time tracker may

include additional features, such as visualization, configuration, integration, and security. Visualization features provide a dashboard or other visual interface to display data about idle time and resource utilization. Configuration features allow administrators to configure thresholds and rules for identifying idle time and triggering actions. Integration features allow the tool to integrate with other monitoring or management tools, such as cloud management platforms or virtualization management software. Finally, security features ensure that the tool is secure and does not expose sensitive data or allow unauthorized access to virtual machines. Overall, the technological overview of a virtual machine idle time tracker is focused on optimizing resource utilization, reducing energy consumption, and improving performance in cloud computing environments. By automating the detection of idle time and taking appropriate actions, virtual machine idle time trackers can help ensure that resources are allocated efficiently and effectively.

Chapter 5

METHODOLOGY & ARCHITECTURE

Architecture and methodology used in virtual machine idle time tracker

The architecture and methodology used in the virtual machine idle time tracker project will depend on the specific requirements and constraints of the project, but here is a general overview of a possible architecture and methodology:

5.1 Architecture:

Virtualization layer: The virtualization layer will be responsible for hosting the virtual machines that will be monitored for idle time. The virtualization platform used could be VMware, VirtualBox, KVM, or Hyper-V.

Monitoring agent: A monitoring agent will be installed on each virtual machine to collect information about the VM's idle time. The agent could be a script running on the VM or a small software agent installed on the VM.

Data collection and storage layer: The data collected by the monitoring agents will be sent to a centralized data collection and storage layer. The data could be stored in a database such as MySQL, PostgreSQL, or MongoDB.

Analytics and reporting layer: The analytics and reporting layer will process the data collected by the monitoring agents to generate reports and alerts. The layer could use tools such as Apache Spark, Hadoop, or Elasticsearch.

User interface layer: The user interface layer will provide a web-based interface for users to view reports and alerts generated by the analytics and reporting layer. The user interface could be developed using technologies such as HTML, CSS, JavaScript, and a frontend framework like React or Angular.

The architecture of a virtual machine idle time tracker can vary depending on the specific implementation and the technologies used. However, a typical architecture

might include the following components:

Data collection: The first component of the architecture is responsible for collecting data about the virtual machines. This can include metrics such as CPU utilization, memory usage, network traffic, and disk I/O.

Data processing: Once data is collected, the next component of the architecture is responsible for processing the data to identify idle time. This can involve applying rules or algorithms to the data, such as monitoring CPU utilization or network traffic over a period of time.

Alerting: Once idle time is identified, the third component of the architecture can generate alerts to notify administrators or other stakeholders. These alerts might include information about the virtual machine that is idle, the amount of idle time, and any recommended actions.

Action execution: Finally, the fourth component of the architecture is responsible for executing actions based on the detected idle time. These actions might include shutting down the virtual machine, migrating it to another host, or reallocating resources to other virtual machines.

In addition to these core components, a virtual machine idle time tracker architecture might also include additional features such as:

Visualization: Providing a dashboard or other visual interface to display data about idle time and resource utilization.

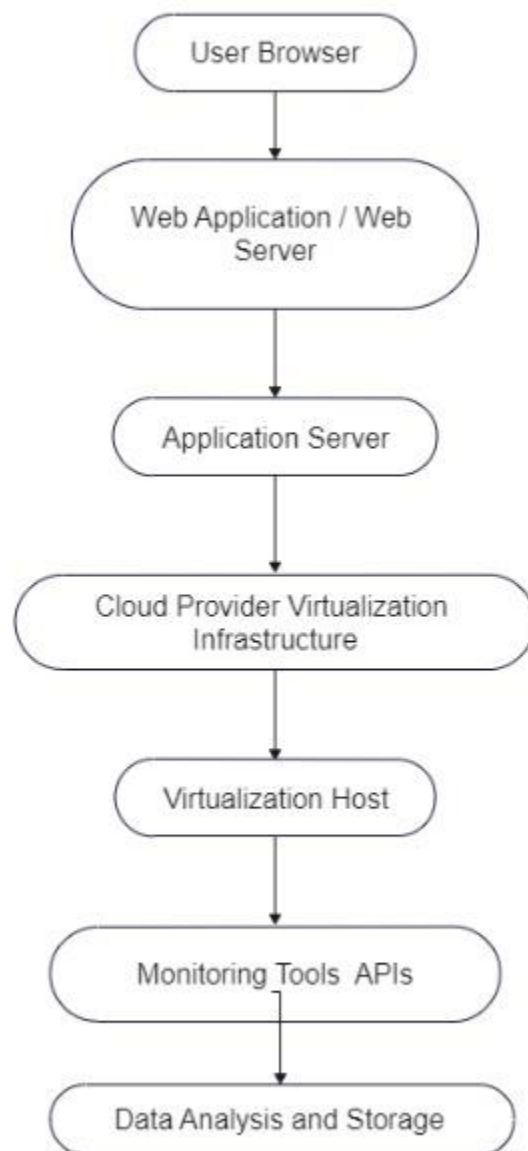
Configuration: Allowing administrators to configure thresholds and rules for identifying idle time and triggering actions.

Integration: Integrating with other monitoring or management tools, such as cloud management platforms or virtualization management software.

Security: Ensuring that the virtual machine idle time tracker is secure and does not expose sensitive data or allow unauthorized access to virtual machines.

The specific architecture of a virtual machine idle time tracker will depend on the requirements of the environment in which it is deployed. However, a well-designed architecture should be scalable, flexible, and modular, allowing it to adapt to changing requirements and integrate with other tools and technologies as needed.

Architecture Diagram:-



5.2 Methodology:

Requirements gathering: The project team will gather requirements from stakeholders to determine the specific features and functionalities needed for the virtual machine idle time tracker.

Design and planning: The team will create a design and plan for the architecture, data model, and user interface of the project.

Implementation: The team will implement the virtualization layer, monitoring agent, data collection and storage layer, analytics and reporting layer, and user interface layer.

Testing: The team will perform testing on each layer of the project to ensure that it is functioning as expected.

Deployment: Once testing is complete, the team will deploy the virtual machine idle time tracker to the production environment.

Maintenance and support: The team will provide ongoing maintenance and support for the virtual machine idle time tracker, including bug fixes and updates to keep the project up-to-date with the latest technologies and security vulnerabilities.

The methodology of a virtual machine idle time tracker typically involves several steps, including data collection, data processing, alerting, and action execution. Here's a more detailed look at each of these steps:

Data collection: The first step in the methodology is to collect data about the virtual machines in the environment. This might involve gathering metrics such as CPU utilization, memory usage, network traffic, and disk I/O. The data can be collected using a variety of tools and techniques, such as performance monitoring agents, log files, or APIs provided by the virtualization platform.

Data processing: Once data is collected, the next step is to process the data to

identify idle time. This can involve applying rules or algorithms to the data, such as monitoring CPU utilization or network traffic over a period of time. For example, a rule might state that a virtual machine is considered idle if CPU utilization is below 10% for more than 30 minutes.

Alerting: Once idle time is identified, the next step is to generate alerts to notify administrators or other stakeholders. These alerts might include information about the virtual machine that is idle, the amount of idle time, and any recommended actions. The alerts can be generated using a variety of mechanisms, such as email, SMS, or integration with a ticketing system.

Action execution: Finally, once an alert is generated, the virtual machine idle time tracker can execute actions based on the detected idle time. These actions might include shutting down the virtual machine, migrating it to another host, or reallocating resources to other virtual machines. The specific action taken will depend on the policies and rules configured for the environment.

In addition to these core steps, the methodology for a virtual machine idle time tracker might also include other elements, such as:

Threshold configuration: The ability to configure thresholds and rules for identifying idle time and triggering actions.

Visualization: The ability to display data about idle time and resource utilization in a dashboard or other visual interface.

Integration: The ability to integrate with other monitoring or management tools, such as cloud management platforms or virtualization management software.

Security: Ensuring that the virtual machine idle time tracker is secure and does not expose sensitive data or allow unauthorized access to virtual machines.

Overall, the methodology of a virtual machine idle time tracker should be designed to optimize resource utilization, reduce energy consumption, and improve performance in cloud computing environments.

Chapter 6

RESULTS & OBSERVATION

6.1 Results and observation

The results and observations of the virtual machine idle time tracker project will depend on the specific implementation and use case, but here are some possible outcomes:

Increased visibility into virtual machine usage: The virtual machine idle time tracker provides insights into how virtual machines are being used, which can help organizations optimize their use of virtual resources.

Reduced costs: By identifying idle virtual machines, organizations can reduce their resource usage and save on costs associated with running unnecessary virtual machines.

Improved performance: By reducing the number of idle virtual machines, the virtual machine idle time tracker can help improve the overall performance of the virtualization platform.

More efficient resource allocation: The data collected by the virtual machine idle time tracker can be used to inform resource allocation decisions, allowing organizations to allocate resources more efficiently based on actual usage patterns.

Better compliance and security: By monitoring virtual machine usage, organizations can ensure compliance with regulatory requirements and improve their security posture by identifying potential security risks.

The results and observations of a virtual machine idle time tracker can provide valuable insights into resource utilization and performance in cloud computing environments. By monitoring and analyzing virtual machines, the tool can identify periods of idle time and take actions to optimize resource utilization and improve

performance. Some of the key results and observations that may be obtained from a virtual machine idle time tracker include:

A virtual machine idle time tracker can identify virtual machines that are not being used and are consuming resources unnecessarily. By identifying these idle virtual machines, administrators can take action to shut them down or reallocate resources to other virtual machines.

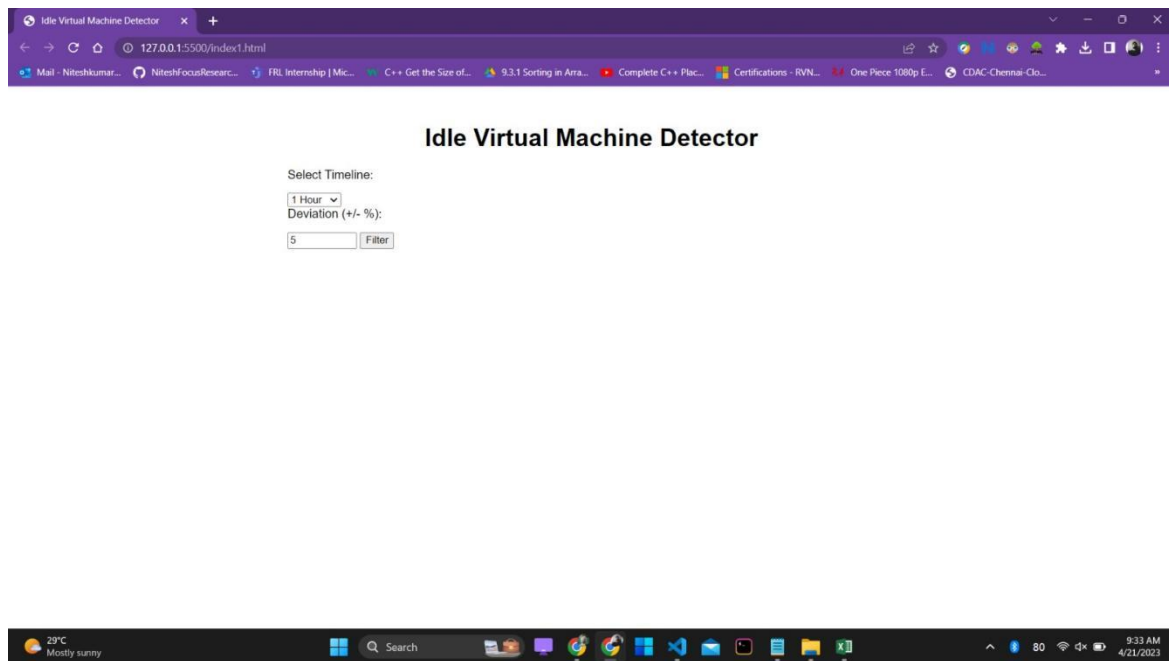
By automating the detection of idle time, a virtual machine idle time tracker can optimize resource utilization in the cloud environment. This can help to reduce energy consumption, improve performance, and save costs.

By taking actions to optimize resource utilization, a virtual machine idle time tracker can help to improve the performance of virtual machines. This can result in faster response times, better user experience, and increased productivity.

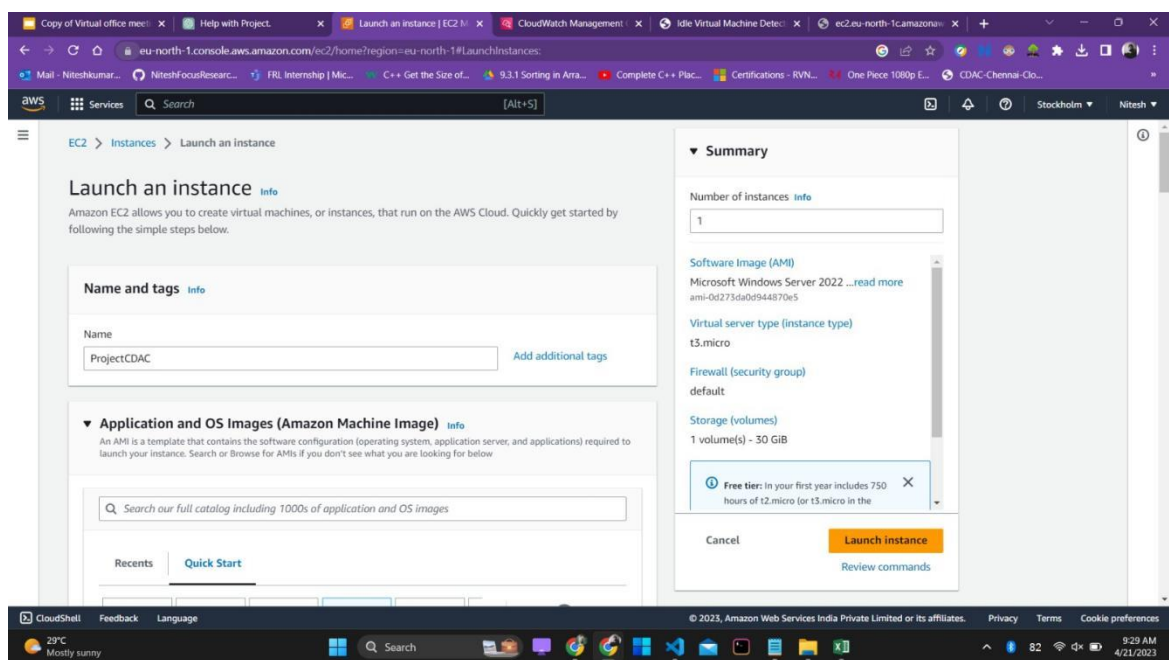
By reducing energy consumption and optimizing resource utilization, a virtual machine idle time tracker can help to save costs in cloud computing environments. This can be particularly important for organizations that operate at scale and have large numbers of virtual machines.

Enhanced visibility: A virtual machine idle time tracker can provide enhanced visibility into resource utilization and performance in the cloud environment. This can help administrators to identify trends, patterns, and areas for improvement, and to make data-driven decisions about resource allocation and performance optimization.

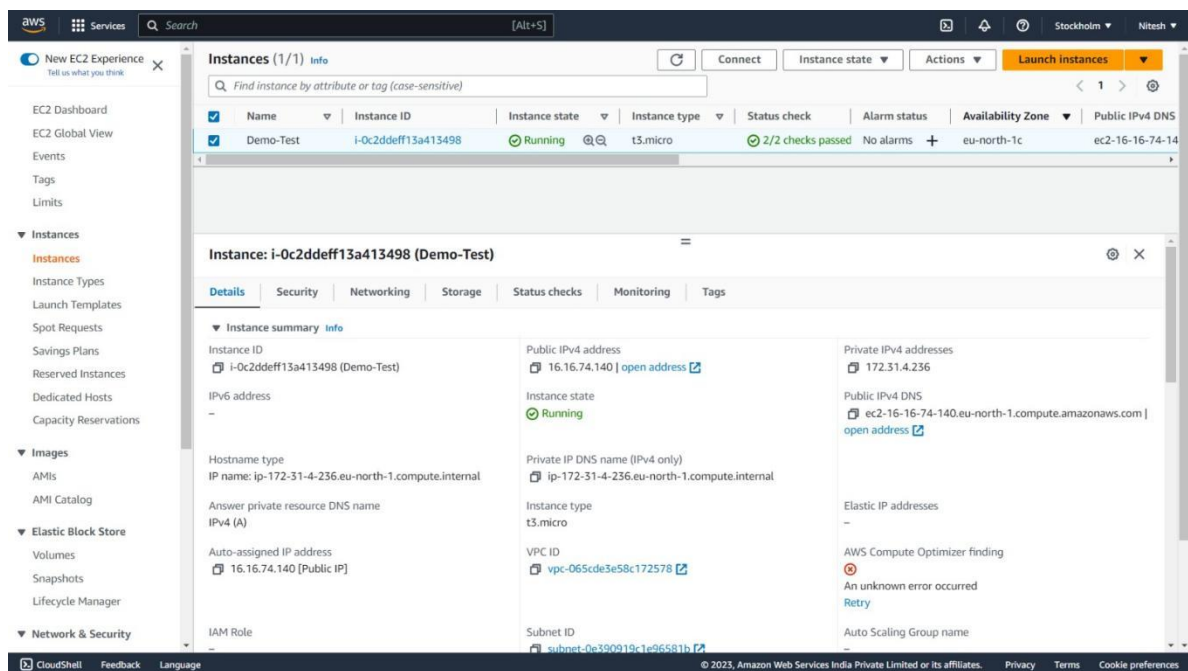
Overall, the results and observations of a virtual machine idle time tracker can help organizations to optimize resource utilization, reduce costs, and improve performance in cloud computing environments. By automating the detection of idle time and taking appropriate actions, virtual machine idle time trackers can help ensure that resources are allocated efficiently and effectively, leading to a more efficient and productive cloud computing environment.



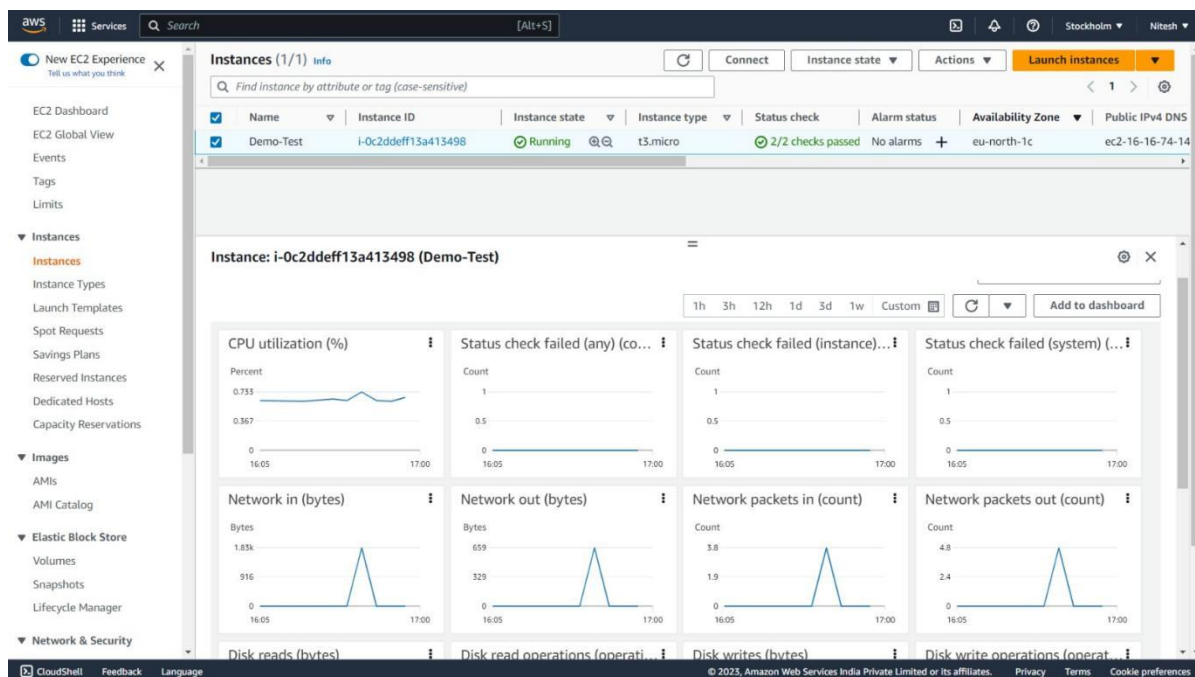
6.1 Web Interface



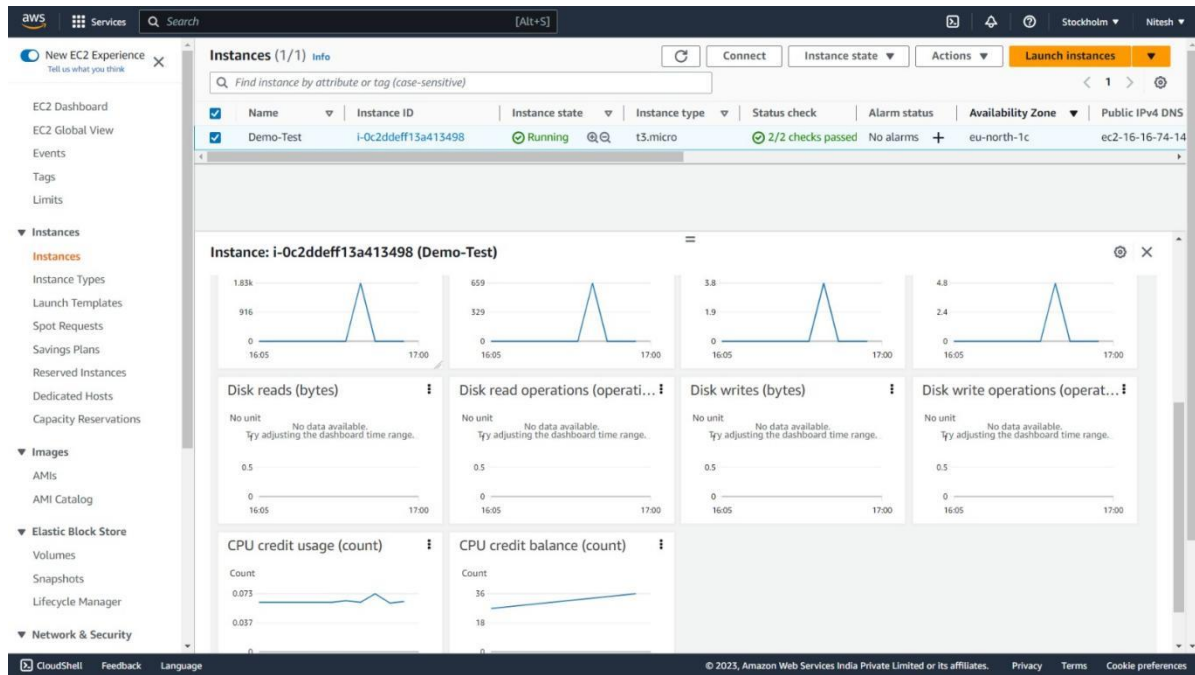
6.2 Launch an VM Instance



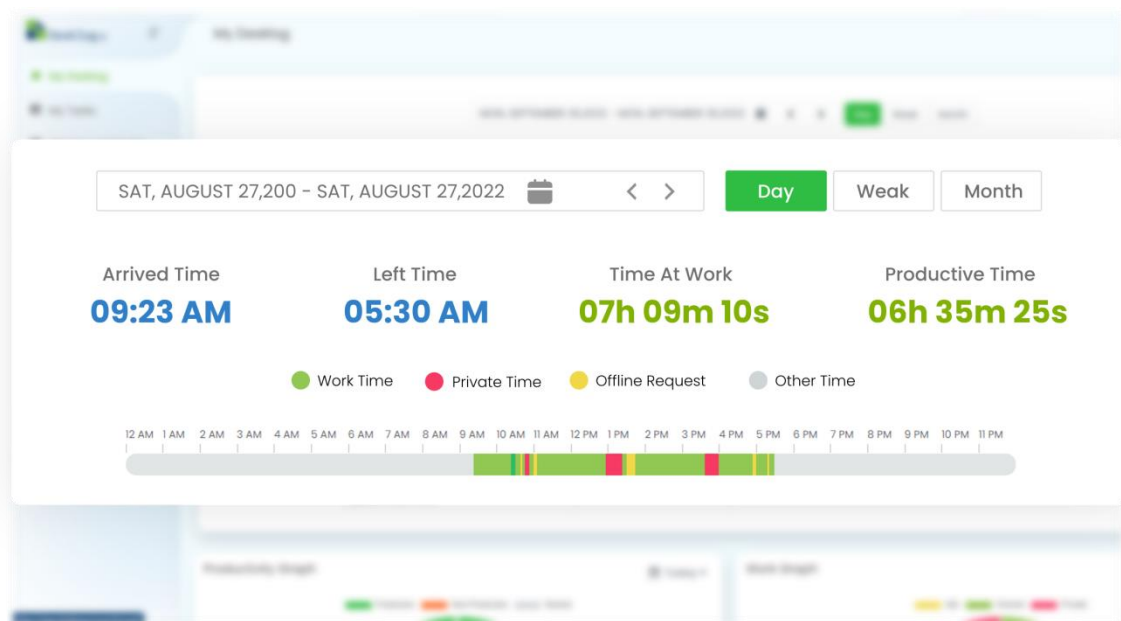
6.3 Success Deployment of VM



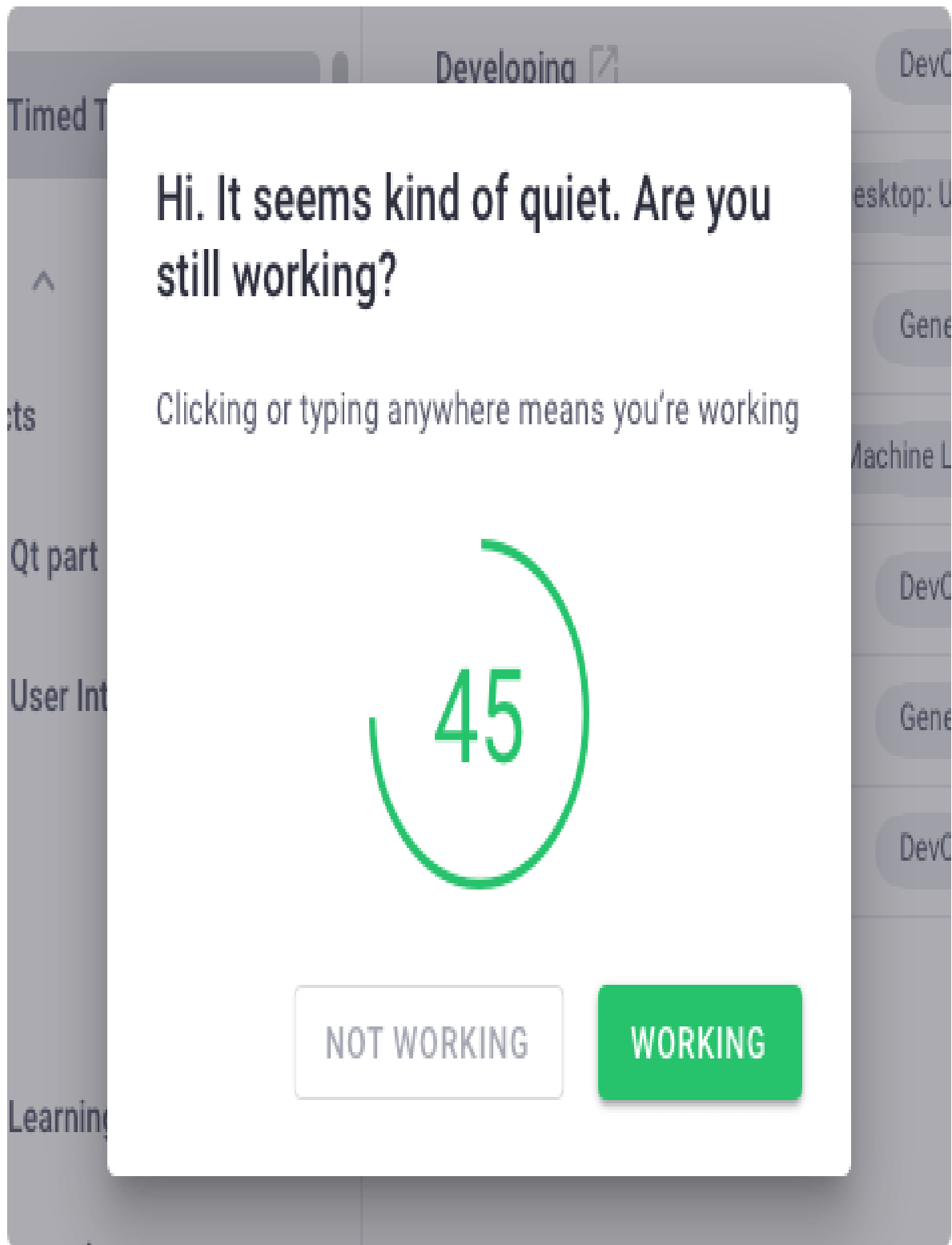
6.4 Information of Various Instance



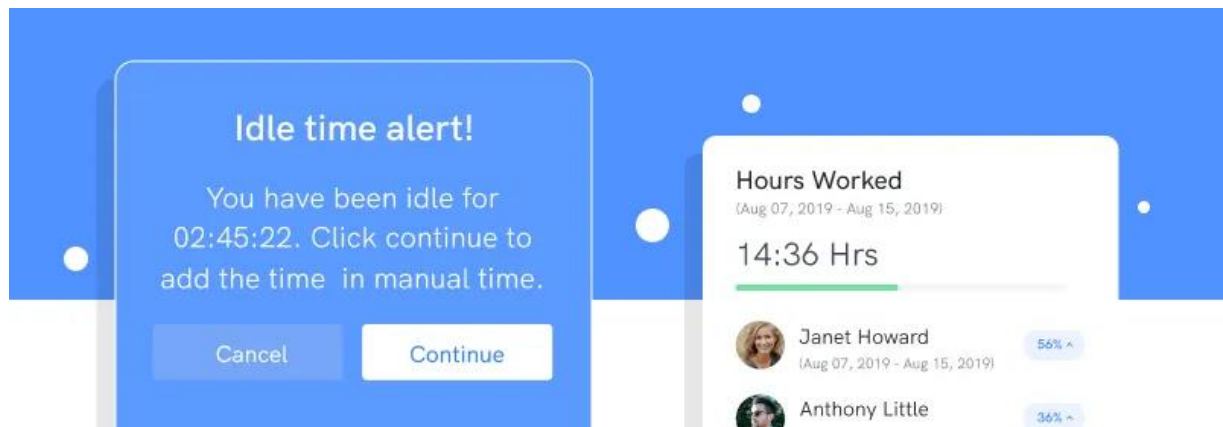
6.5 Information of Various Instances



6.6 Time Tracking Interface



6.7 Monitoring Result



6.8 Idle time alerts



6.9 Utilization in given time period



6.10 Final Result



```

1  import boto3
2
3  # create a CloudWatch client with AWS access keys
4  cloudwatch = boto3.client(
5      'cloudwatch',
6      region_name='us-east-1',
7      aws_access_key_id='AKIAQTHSSGWZT64YGXBC',
8      aws_secret_access_key='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'
9  )
10
11 # specify the metric and time range you want to retrieve data for
12 metric_name = 'CPUUtilization'
13 namespace = 'AWS/EC2'
14 start_time = '2023-04-01T00:00:00Z'
15 end_time = '2023-04-24T00:00:00Z'
16
17 # retrieve the metric data
18 response = cloudwatch.get_metric_data(
19     MetricDataQueries=[
20         {
21             'Id': 'm1',
22             'MetricStat': {
23                 'Metric': {
24                     'Namespace': namespace,
25                     'MetricName': metric_name
26                 },
27                 'Period': 300,
28                 'Stat': 'Average',
29                 'Unit': 'Percent'
30             },
31             'ReturnData': True
32         }
33     ],
34     StartTime=start_time,
35     EndTime=end_time
36 )
37
38 # print the metric data
39 print(response)
40

```

6.11 Boto py code

```

1 import boto3
2
3
4 access_key = 'AKIAQTHSSGWZT64YGXBC'
5 secret_key = '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'
6 region = 'eu-north-1'
7
8 ec2 = boto3.client('ec2', region_name=region, aws_access_key_id=access_key, aws_secret_access_key=secret_key)
9 cloudwatch = boto3.client('cloudwatch', region_name=region, aws_access_key_id=access_key, aws_secret_access_key=secret_key)
10
11 instance_id = 'i-0c2ddef13a413498'
12
13 response = cloudwatch.get_metric_data(
14     MetricDataQueries=[
15         {
16             'Id': 'cpu',
17             'MetricStat': {
18                 'Metric': {
19                     'Namespace': 'AWS/EC2',
20                     'MetricName': 'CPUUtilization',
21                     'Dimensions': [
22                         {
23                             'Name': 'InstanceId',
24                             'Value': instance_id
25                         }
26                     ],
27                 },
28                 'Period': 300,
29                 'Stat': 'Average',
30                 'Unit': 'Percent'
31             },
32             'ReturnData': True
33         }
34     ],
35     StartTime='2023-04-24T00:00:00Z',
36     EndTime='2023-04-24T23:59:59Z'
37 )
38
39 datapoints = response['MetricDataResults'][0]['Values']
40 print(datapoints)
41
42 def is_instance_idle(instance_id, start_time, end_time):
43     response = ec2.get_metric_data(
44         MetricDataQueries=[
45             {
46                 'Id': 'cpu',
47                 'MetricStat': {
48                     'Metric': {
49                         'Namespace': 'AWS/EC2',
50                         'MetricName': 'CPUUtilization',
51                         'Dimensions': [
52                             {
53                                 'Name': 'InstanceId',
54                                 'Value': instance_id
55                             }
56                         ],
57                     },
58                     'Period': 300,
59                     'Stat': 'Average',
60                     'Unit': 'Percent'
61                 },
62                 'ReturnData': True
63             }
64         ],
65         StartTime=start_time,
66         EndTime=end_time
67     )
68
69     datapoints = response['MetricDataResults'][0]['Values']
70     avg_cpu_utilization = sum(datapoints) / len(datapoints)
71     return abs(avg_cpu_utilization - 0.0) <= 0.05
72
73 from datetime import datetime, timedelta
74
75 def get_idle_instances(timeline):
76     now = datetime.utcnow()
77     end_time = now.strftime('%Y-%m-%dT%H:%M:%SZ')
78     start_time = (now - timedelta(hours=int(timeline[:-1]))).strftime('%Y-%m-%dT%H:%M:%SZ')
79
80     response = ec2.describe_instances()
81
82     instances = []
83     for reservation in response['Reservations']:
84         for instance in reservation['Instances']:
85             if instance['State']['Name'] == 'running' and is_instance_idle(instance['InstanceId'], start_time, end_time):
86                 instances.append(instance)
87     return instances
88

```

6.12 Interface code

REFERENCES

1. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.
2. S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors," in *ACM SIGOPS Operating Systems Review*, vol. 41, no. 3. ACM, 2007, pp. 275–287.
3. Z. J. Estrada, F. Deng, Z. Stephens, C. Pham, Z. Kalbarczyk, and R. Iyer, "Performance comparison and tuning of virtual machines for sequence alignment software," *Scalable Computing: Practice and Experience*, vol. 16, no. 1, 2015.
4. Z. J. Estrada, Z. Stephens, C. Pham, Z. Kalbarczyk, and R. K. Iyer, "A performance evaluation of sequence alignment software in virtualized environments," in *Cluster, Cloud and Grid Computing (CCGrid)*, 2014 14th IEEE/ACM International Symposium on. IEEE, 2014, pp. 730–737.
5. Z. J. Estrada, Z. Stephens, C. Pham, Z. Kalbarczyk, and R. K. Iyer, "A performance evaluation of sequence alignment software in virtualized environments," in *Cluster, Cloud and Grid Computing (CCGrid)*, 2014 14th IEEE/ACM International Symposium on. IEEE, 2014, pp. 730–737.
6. A. Madhavapeddy, R. Mortier, C. Rotsos, D. Scott, B. Singh, T. Gazagnaire, S. Smith, S. Hand, and J. Crowcroft, "Unikernels: Library operating systems for the cloud," in *ACM SIGPLAN Notices*, vol. 48, no. 4. ACM, 2013, pp. 461–472.