

(Cyber Security)



23CY504 – CLOUD SECURITY

(REGULATION 2022)

LAB MANUAL

Laboratory In-charge
Ms.S.Priyadharshini
Assistant Professor

PREFACE

In the ever-evolving landscape of technology, data has emerged as one of the most valuable resources of the 21st century. The term "Big Data" encapsulates the enormous volumes of structured and unstructured data generated at unprecedented rates, coming from various sources such as social media, sensors, transactions, and more. The real challenge lies not only in capturing and storing this data but in analyzing it to extract meaningful insights that can drive decision-making and innovation.

The field of Big Data Analytics is at the forefront of this revolution, offering powerful tools and techniques to process, analyze, and visualize large datasets. This lab manual is designed to guide students through the practical aspects of Big Data Analytics, providing hands-on experience with industry-standard tools and methodologies.

The primary goal of this lab is to equip students with the skills necessary to handle real-world data analytics challenges. Through a series of structured exercises and projects, students will learn Understand the Fundamentals, Utilize Analytical Tools, Apply Machine Learning Techniques, Implement Data Visualization, Develop Real-world Applications.

The exercises in this lab manual are designed to be both challenging and rewarding, encouraging students to think critically and creatively. By the end of this course, students will not only be proficient in the technical aspects of Big Data Analytics but also be prepared to leverage these skills in their future careers.

FACULTY OF COMPUTER SCIENCE AND ENGINEERING

SRI KRISHNA COLLEGE OF TECHNOLOGY

COIMBATORE – 641 042

Prepared by
S.Priyadharshini
Assistant Professor
/IOT

Verified by
Dr. Suma Sira Jacob
Associate Professor
PC/ CSE (AIML CYS & IoT)

PROFILE OF THE INSTITUTION

Nestled at the foothills of the Western Ghats, located in a sprawling 52-acre campus in Kovaipudur, Coimbatore, Sri Krishna College of Technology (SKCT) is a vibrant institute of higher education established in 1985 promoted by Sri Krishna Institutions. An extraordinary freedom of opportunity—to explore, to collaborate and to challenge oneself is the hallmark of the Institute. Being an autonomous institute, affiliated to Anna University, Chennai, and approved by AICTE, New Delhi, SKCT lays strong emphasis on collaborative research and stands apart from other institutes by its participatory work culture, student care Programmes and high industry interaction.

In a span of 38 years, it has emerged as one of the premier engineering colleges for learning, discovery and innovation due to the dynamic leadership of the Chairperson and Managing Trustee Smt. S. Malarvizhi. Being an acclaimed educationalist, she continues to contribute profusely for the glory and happiness of advancing generations. The college is accredited with A Grade by NAAC and eligible undergraduate programs are accredited by the National Board of Accreditation (NBA), New Delhi. The college offers 11 undergraduate Programmes, 6 Postgraduate Programmes and 5 Doctorial Programmes in Engineering, Technology, and Management Studies.

VISION:

Sri Krishna College of Technology aspires to be recognized as one of the pioneers in imparting world class technical education through technology enabled innovative teaching learning processes with a focus on research activities to cater, to the societal needs.

MISSION:

To be recognized as centre of excellence in science, engineering and technology through effective teaching and learning processes by providing a conducive learning environment.

To foster research and development with creative and entrepreneurial skills by means of innovative applications of technology. Accomplish expectations of the society and industry by nurturing the students to be competent professionals with integrity.

COURSES OFFERED

UNDER GRADUATE PROGRAMMES (Four Years B.E / B.Tech)

- B.E - Civil Engineering
- B.E - Computer Science and Engineering
- B.E - Computer Science and Engineering (Cyber Security)
- B.E - Computer Science and Engineering (Internet of Things)
- B.E - Computer Science and Engineering (Artificial Intelligence and Machine Learning)
- B.E - Electronics and Communication Engineering
- B.E - Electrical and Electronics Engineering
- B.E - Instrumentation and Control Engineering
- B.E - Mechanical Engineering
- B.Tech - Artificial Intelligence and Data Science
- B.Tech - Information Technology

POST GRADUATE PROGRAMMES (Two Years)

- Master of Business Administration
- M.E - Applied Electronics
- M.E – Computer Science Engineering
- M.E –Engineering Design
- M.E - Power System Engineering
- M.E - Structural Engineering

DOCTORAL PROGRAMMES (Ph.D.)

- Civil Engineering
- Computer Science and Engineering
- Electronics and Communication Engineering
- Electrical and Electronics Engineering
- Mechanical Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING (CYS)

The Department of Computer Science and Engineering (CYS) at the esteemed institution, SKCT, which was established in 2022. At our institution, we are committed to providing a 4-year Bachelor of Engineering (B.E.) degree with a specific focus on Internet of Things. The eminent team of faculty is dedicated in delivering a high-quality education to equip students with the necessary skills to navigate the dynamic and ever-changing domains of AIML and IoT. The program has been strategically developed to cultivate creativity, critical thinking, and problem-solving aptitudes, equipping our graduates with the necessary capabilities to contribute sustainable solutions to industrial and society problems.

VISION:

The department of CSE fosters a conducive ambience to meet the global standards by equipping the students with modern techniques in the area of Computer Science and relevant research to address the societal needs.

MISSION:

- To provide positive working environment that would help the students perform to their highest abilities in various fields of computer science.
- To enable students and faculty with the best of technologies and knowledge emerging in the domain of Computer Science and Engineering.
- To establish nationally and internationally recognized research centers and expose the students to broad research experience.

PROGRAM EDUCATIONAL OBJECTIVES:

PEO1: Apply the acquired engineering knowledge to solve economic, social, ethical, and environmental issues related to Internet of Things.

PEO2: Adapt the emerging Information and Communication Technologies to innovate and to cater the Industrial and Societal needs.

PEO3: Contribute Internet of Things expertise to research and development and create novel products that benefit society.

PROGRAM OUTCOMES:

PO 1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO 2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO 3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO 4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the Information to provide valid conclusions.

PO 5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO 6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO 7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO 8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO 9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO 10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO 11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO 12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

PSO1: Acquire and apply standard Software Engineering practices and strategies in IoT project development to deliver a quality product for industry success.

PSO2: Analyze connected sensors, devices and equipment for transferring data over a network.

TABLE OF CONTENT

Experiment No:	NAME OF THE EXPERIMENT
1	Installation of VirtualBox/VMware Workstation with Multiple OS Flavours on Windows 7/8
2	Installation of a C Compiler in a Virtual Machine Using Virtual Box and Execution of Simple C Programs
3	Use Google App Engine (GAE) Launcher to Deploy Web Applications
4	Install google app engine create hello world app and other simple web application using java and python
5	Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim
6	Find a procedure to transfer the files from one virtual machine to another virtual machine.
7	Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version).
8	Install Hadoop single node cluster and run simple applications like word count
9	Innovative Approaches

RUBRICS

Items	Excellent	Good	Satisfactory	Needs Improvement
BACKGROUND THEORY (PRE LAB) (20 MARKS)	15-20	10-14	5-9	0-4
	Objective and Procedure are highly /maximally efficient and effective,demonstrating strong understanding of sequence	Objective and Procedure are efficient and effective, demonstrating moderate understanding of sequence	Objective and Procedure are somewhat efficient and effective, demonstrating adequate understanding of sequence.	Objective and Procedure are inefficient and/or ineffective, demonstrating limited understanding of sequence.
PROCEDURE (EXECUTION) (40 MARKS)	34-40	27-33	21-32	0-20
	The design uses appropriate Syntax and Structures. The program overall design is appropriate.	The design generally uses appropriate structures. Program elements exhibit good design.	Not all of the selected structures are appropriate. Some of the elements are appropriately designed.	Few of the selected structures are appropriate. Structures are not well designed.
DISCUSSION (EXECUTION) (20 MARKS)	15-20	10-14	5-9	0-4
	Design contains no evidence of misunderstanding or misinterpreting. Functionality produces correct answers or appropriate results for all inputs tested.	Design is free from major misunderstandings, but may contain non-standard usage or superfluous elements. Functionality produces correct answers or appropriate results for most inputs	Design contains errors that signal misunderstanding of syntax. Functionality approaches correct answers or appropriate results for most inputs, but can contain miscalculations in some cases	Design does not compile or contains typographical errors leading to undefined names. Functionality does not produce correct answers or appropriate results for most inputs.
CONCLUSION &VIVAVOCE (POST LAB) (20 MARKS)	15-20	10-14	5-9	0-4
	Clearly and effectively documented including descriptions of all class variables. Specific purpose noted for each function, control structure, input requirements, and output results. Masterfully defends by providing clear and insightful answers to questions	Clearly documented including descriptions of all class variables. Specific purpose is noted for each function and control structure. Competently defends by providing very helpful answers	Basic documentation has been completed including descriptions of all class variables. Purpose is noted for each function. Answers questions, but often with little insight	Very limited or no documentation included. Documentation does not help the reader understand the code. Very less answers / Does not answer

Experiment No 1 :Installation of VirtualBox/VMware Workstation with Multiple OS Flavours on Windows 7/8

AIM:

To install VirtualBox or VMware Workstation on Windows 7/8 and set up virtual machines with different flavours of Linux (e.g., Ubuntu, Fedora) or Windows OS.

BACKGROUND THEORY:

Virtualization allows multiple operating systems to run on a single physical machine by sharing hardware resources. Two main software used for virtualization are:

- Oracle VirtualBox (Free, Open-source)
- VMware Workstation (Proprietary, Paid with a free version available - VMware Workstation Player)

Benefits of Virtualization:

- Run multiple OSs simultaneously
- Ideal for testing, development, and educational use
- Isolation between guest and host OS
- No need for dual booting

Software Required:

S. No	Software	Version
1	Host OS (Windows 7/8)	64-bit
2	Oracle VirtualBox or VMware	Latest
3	ISO File for Linux/Windows	Ubuntu 22.04 / Fedora / Windows 10 ISO

Algorithm / Steps:

Step 1: Install Virtualization Software

1. Download **VirtualBox** from <https://www.virtualbox.org>
OR
Download **VMware Workstation Player** from <https://www.vmware.com>

2. Run the installer and follow the on-screen instructions to complete the installation.

Step 2: Create a New Virtual Machine

1. Open VirtualBox or VMware.
2. Click **New VM**.
3. Give a name (e.g., Ubuntu_VM), select **type** (Linux/Windows), and **version**.
4. Allocate RAM (Recommended: 2048 MB for Linux, 4096 MB for Windows).
5. Create a virtual hard disk (10–30 GB recommended).
6. Choose hard disk file type (VDI for VirtualBox or VMDK for VMware).
7. Proceed with a dynamically allocated or fixed size disk.

Step 3: Mount ISO and Start Installation

1. In VM settings, mount the downloaded **ISO file** under "Optical Drive".
2. Start the virtual machine.
3. Follow the OS installation steps (like a real PC).

Step 4: Post-Installation Setup

1. Install **Guest Additions (VirtualBox)** or **VMware Tools** for better performance and shared clipboard.
2. Customize screen resolution, networking, and shared folders if needed.

Code

Create VM

```
VBoxManage createvm --name "UbuntuVM" --ostype Ubuntu_64 --register
```

Set memory and boot

```
VBoxManage modifyvm "UbuntuVM" --memory 2048 --boot1 dvd --nic1 nat
```

Create virtual disk

```
VBoxManage createhd --filename "UbuntuVM.vdi" --size 20000
```

Attach storage

```
VBoxManage storagectl "UbuntuVM" --name "SATA Controller" --add sata --controller  
IntelAhci
```

```
VBoxManage storageattach "UbuntuVM" --storagectl "SATA Controller" --port 0 --device 0  
--type hdd --medium "UbuntuVM.vdi"
```

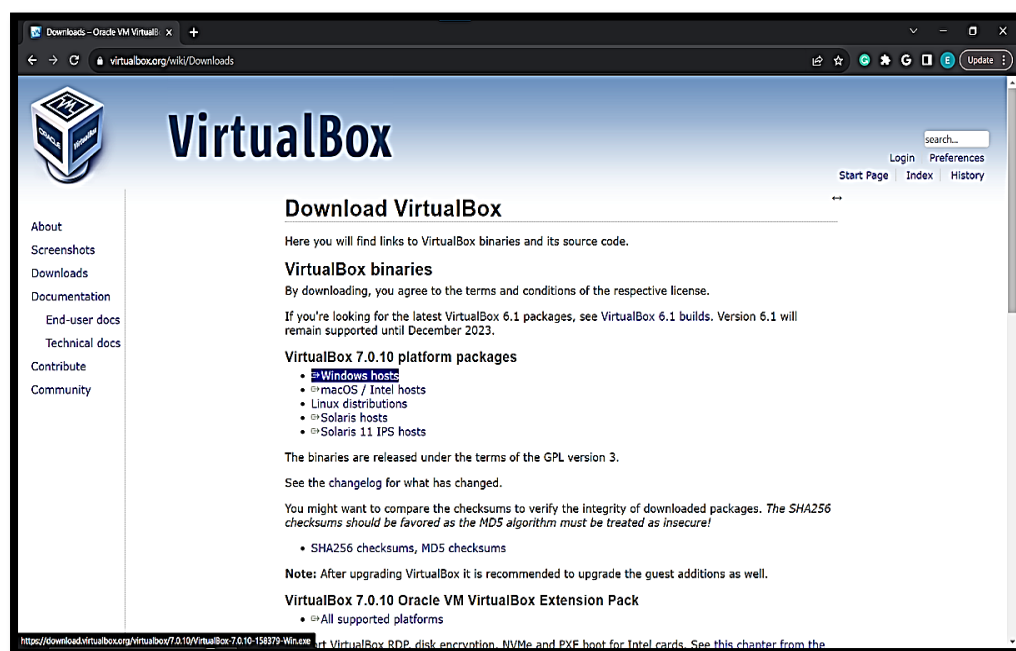
```
VBoxManage storageattach "UbuntuVM" --storagectl "SATA Controller" --port 1 --device 0  
--type dvddrive --medium "ubuntu-22.04.iso"
```

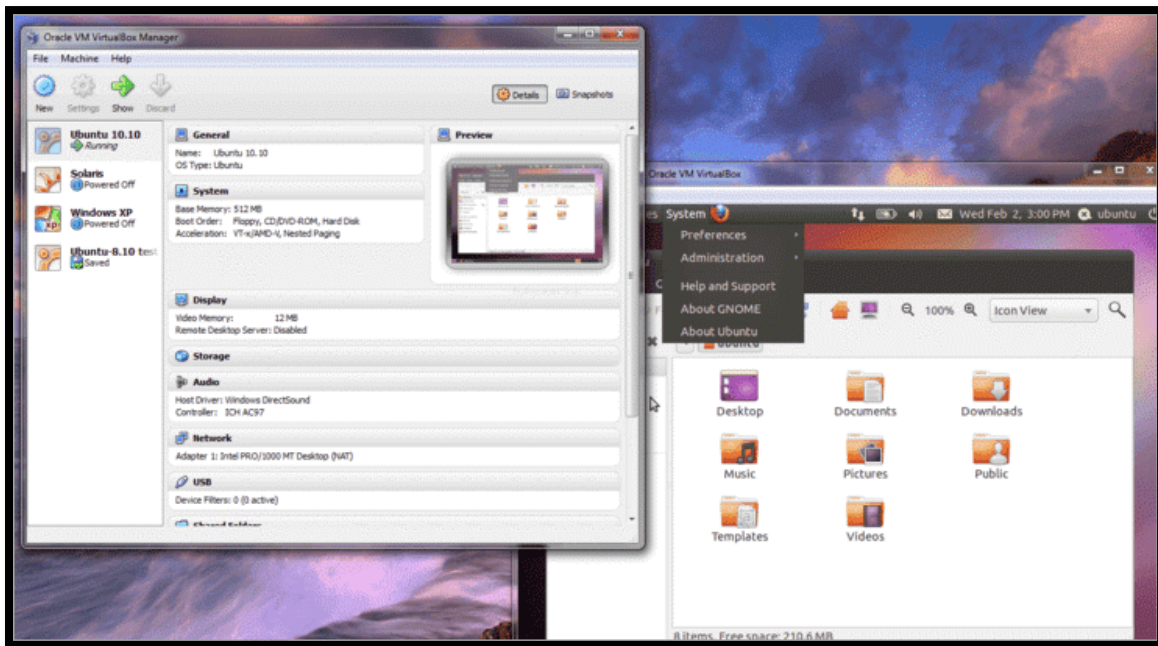
```
# Start VM
```

```
VBoxManage startvm "UbuntuVM"
```

Screenshots

1. VirtualBox/VMware installation screen
2. New VM creation steps
3. ISO mounting and OS installation
4. Running guest OS inside the VM





Result:

Successfully installed and configured VirtualBox/VMware Workstation on Windows 7/8 and ran multiple virtual machines with different OS flavours like Ubuntu and Windows 10.

Pre-Viva Questions

1. State the purpose of using virtualization tools like VirtualBox or VMware.
2. Mention the advantages of installing multiple OS flavours on a single host system.
3. Describe the difference between a host OS and a guest OS.
4. List common OS flavours that can be installed on virtual machines.
5. Explain the basic system requirements for running a virtual machine efficiently.
6. Identify the steps involved in creating a new virtual machine.

Post-Viva Questions

1. Share your experience in installing and configuring a guest operating system.
2. Describe the settings you configured while creating the virtual machine (e.g., RAM, disk).
3. Mention how you enabled networking in the guest OS.
4. Explain how multiple OSes can coexist without affecting the host machine.
5. Suggest use cases where virtual machines are better than dual boot.
6. Reflect on any difficulties you faced during installation and how you resolved them.

Experiment No 2: Installation of a C Compiler in a Virtual Machine Using Virtual Box and Execution of Simple C Programs

AIM:

To install a C compiler (such as `gcc`) in a Linux virtual machine created using VirtualBox and execute basic C programs.

BACKGROUND THEORY:

The C programming language is a foundational language for system and application development. To compile and run C programs in a Linux environment, we use the GCC (GNU Compiler Collection).

WHAT IS GCC?

- GCC is a compiler system produced by the GNU Project supporting various programming languages.
- It is the standard compiler for most Unix-like systems.

VirtualBox + Linux + GCC = Portable Development Environment

Using a VM helps you test, code, and debug in a safe, isolated, and consistent environment.

SOFTWARE REQUIRED:

S. No	Software	Version
1	Oracle VirtualBox	Latest
2	Ubuntu / Fedora ISO	Ubuntu 22.04 LTS
3	GCC Compiler	Preinstalled or installable via package manager
4	Terminal / Code Editor	Gedit, nano, or VS Code (optional)

ALGORITHM / STEPS:

Step 1: Start Your Linux VM

- Open VirtualBox.
- Select your Linux virtual machine and click **Start**.

Step 2: Open Terminal

- Once the Linux desktop loads, press Ctrl + Alt + T to open the terminal.

Step 3: Update Package Repository

sudo apt update

For Fedora:

sudo dnf update

Step 4: Install GCC Compiler

For Ubuntu/Debian-based systems:

sudo apt install build-essential

For Fedora-based systems:

sudo dnf groupinstall "Development Tools"

Step 5: Verify GCC Installation

gcc --version

Output should show installed version like:

gcc (Ubuntu 11.4.0) 11.4.0

Step 6: Write and Execute a Simple C Program

Code: Hello World in C

1. Create a C file using any text editor:

nano hello.c

2. Paste the following code:

```
#include <stdio.h>
```

```
int main() {  
    printf("Hello, Virtual Machine World!\n");  
    return 0;  
}
```

3. Save and Exit (Ctrl + O, Enter, then Ctrl + X)

Step 7: Compile the Program

```
gcc hello.c -o hello
```

This will create an output file hello.

Step 8: Run the Executable

```
./hello
```

Expected Output:

Hello, Virtual Machine World!

Output Screenshots :

1. Terminal showing gcc --version
2. Code written in editor (nano or gedit)
3. Output of the compiled C program


```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19041.388]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Source\Programs

C:\Source\Programs>gcc c -o hello_world.exe hello_world.c

C:\Source\Programs>hello_world.exe
```

```
lab@lab: ~
There are 3 choices for the alternative g++ (providing /usr/bin/g++).

  Selection      Path                      Priority    Status
-----
0             /usr/bin/g++-4.7          100         auto mode
1             /usr/bin/g++-4.7          100         manual mode
2             /usr/bin/g++-4.8          40          manual mode
* 3           /usr/bin/g++-4.9          60          manual mode

Press enter to keep the current choice[*], or type selection number:
lab@lab:~$
lab@lab:~$ gcc --version
gcc (GCC) 5.3.0
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

lab@lab:~$ ls /usr/bin/gcc*
/usr/bin/gcc           /usr/bin/gcc-ar-4.8  /usr/bin/gcc-ranlib-4.7
/usr/bin/gcc-4.7       /usr/bin/gcc-ar-4.9  /usr/bin/gcc-ranlib-4.8
/usr/bin/gcc-4.8       /usr/bin/gcc-nm-4.7  /usr/bin/gcc-ranlib-4.9
/usr/bin/gcc-4.9       /usr/bin/gcc-nm-4.8
/usr/bin/gcc-ar-4.7    /usr/bin/gcc-nm-4.9
lab@lab:~$
```

Result:

Successfully installed GCC in the Linux virtual machine, compiled, and executed a basic C program.

Pre-Viva Questions

1. State the steps involved in setting up a virtual machine in VirtualBox.
2. Mention the importance of installing a C compiler inside a virtual environment.
3. Describe common C compilers available for Linux distributions.
4. List Linux commands used to install software packages from the terminal.
5. Identify basic components required to compile and run a C program.
6. Explain why VirtualBox is preferred for OS-level sandboxing.

Post-Viva Questions

1. Share the command used to install the C compiler in your virtual machine.
2. Describe the steps you followed to write, compile, and run a C program.
3. Mention any errors encountered during installation or compilation and how you solved them.
4. Explain the purpose of using `gcc` or `g++` in Linux environments.
5. Reflect on the advantages of testing programs inside a VM instead of the host OS.
6. Suggest how the setup can be extended to support other languages or tools.

Experiment No:3 Installing Google App Engine and Creating Hello World Web Applications in Java and Python

AIM

To install Google App Engine and develop basic web applications such as "Hello World" using both Java and Python, and deploy them using GAE's standard environment.

SOFTWARE REQUIRED

- **Google Cloud SDK** (includes App Engine deployment tools)
- **Python 3.x**
- **Java JDK 11+**
- **Apache Maven** (for Java)
- **Text Editor:** VS Code / Notepad++ / IntelliJ
- **Web Browser** (Chrome or Firefox)
- **Internet Connection**
- **Operating System:** Windows/Linux/Mac

BACKGROUND THEORY

Google App Engine (GAE) is a fully managed serverless platform by Google Cloud. It supports various programming languages like Python, Java, Go, Node.js, etc., and allows you to build scalable web apps and APIs without managing infrastructure.

- **GAE Standard Environment:** Predefined runtimes and sandbox environment
- **GAE Flexible Environment:** Custom runtimes with Docker container support

Both Python and Java web apps can be deployed using GAE by creating an `app.yaml` configuration and using the `gcloud app deploy` command.

PART A: PYTHON – HELLO WORLD APP

ALGORITHM

1. Install Python and Google Cloud SDK
2. Create a GCP project and initialize SDK
3. Build a simple Flask web app

4. Create `app.yaml` file
5. Deploy using `gcloud app deploy`

LONG CODE (Python: `main.py`)

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello():
    return "Hello, World! This is a Python app on GAE."

if __name__ == '__main__':
    app.run()
```

app.yaml

```
runtime: python39
entrypoint: gunicorn -b :$PORT main:app
```

```
handlers:
- url: /*
  script: auto
```

requirements.txt

```
Flask==2.3.3
gunicorn==21.2.0
```

SAMPLE OUTPUT (Python)

When accessed via browser after deployment:

Hello, World! This is a Python app on GAE.

PART B: JAVA – HELLO WORLD APP

ALGORITHM

1. Install Java JDK and Apache Maven
2. Create a Maven project for GAE Java App
3. Add dependencies and servlet code
4. Create `appengine-web.xml` configuration

5. Deploy using gcloud app deploy

CODE (Java: Servlet-based App)

HelloServlet.java

```
import java.io.IOException;
import javax.servlet.http.*;

public class HelloServlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println("Hello, World! This is a Java app on GAE.");
    }
}
```

web.xml

```
<web-app>
  <servlet>
    <servlet-name>HelloServlet</servlet-name>
    <servlet-class>HelloServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>HelloServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>
</web-app>
```

appengine-web.xml

```
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <runtime>java11</runtime>
</appengine-web-app>
```

SAMPLE OUTPUT (Java)

Hello, World! This is a Java app on GAE.

RESULT

Python and Java-based web applications were successfully created and deployed on Google App Engine. The experiment demonstrated deploying scalable cloud applications using standard GAE tools and configurations.

PRE VIVA QUESTIONS

1. State the purpose of using Google App Engine in cloud platforms.
2. Mention supported languages in GAE Standard Environment.
3. Describe the role of the app.yaml or appengine-web.xml file.
4. List tools required to deploy Python and Java apps in GAE.
5. Explain the difference between runtime and entrypoint.
6. Identify the use of gcloud commands in deployment.

POST VIVA QUESTIONS

1. Describe the output observed after deployment.
2. Explain how GAE handles incoming web requests.
3. Share your experience using Python Flask vs Java Servlets on GAE.
4. Mention challenges faced during app deployment and how you resolved them.
5. Suggest extensions like form handling, database integration, or API creation.
6. Evaluate the performance and scalability benefits of serverless deployment.

Experiment 5: Simulate a Cloud Scenario Using CloudSim with a Custom Scheduling Algorithm

AIM

To simulate a cloud environment using CloudSim and implement a custom VM scheduling algorithm not present in the default library.

SOFTWARE REQUIRED

- CloudSim 3.0.3 or later
- Java JDK 8+
- Eclipse IDE
- Windows/Linux/Mac OS

BACKGROUND THEORY

CloudSim is a simulation toolkit for modeling cloud computing environments. It supports modeling of data centers, hosts, VMs, and scheduling policies.

Custom scheduling allows you to override existing policies like TimeShared or SpaceShared by implementing your own logic in `VmScheduler`.

ALGORITHM

1. Set up a datacenter and host.
2. Create a list of VMs.
3. Define a **custom VM scheduler** by extending `VmScheduler`.
4. Deploy cloudlets.
5. Observe performance under the custom policy.

CODE (Custom VmSchedulerRoundRobin.java)

```
public class VmSchedulerRoundRobin extends VmScheduler {  
  
    private int currentIndex = 0;
```

```

public VmSchedulerRoundRobin(List<? extends Pe> pelist) {
    super(pelist);
}

@Override
public boolean allocatePesForVm(Vm vm, List<Double> mipsShare) {
    Map<String, List<Double>> peAllocationMap = getPeAllocationMap();
    List<Pe> peList = getPeList();

    if (currentIndex >= peList.size()) {
        currentIndex = 0;
    }

    List<Double> allocatedMips = new ArrayList<>();
    allocatedMips.add(mipsShare.get(0)); // Single MIPS value
    peAllocationMap.put(vm.getUid(), allocatedMips);

    currentIndex++;
    return true;
}

@Override
public void deallocatePesForVm(Vm vm) {
    getPeAllocationMap().remove(vm.getUid());
}
}

```

SAMPLE OUTPUT

Custom Round Robin VM Scheduling Simulation Started...
 Datacenter created
 VMs allocated in Round Robin manner
 Simulation ended successfully.

RESULT

A custom Round-Robin VM scheduler was successfully implemented and tested using CloudSim, simulating cloud workload management.

PRE VIVA QUESTIONS

1. State the role of CloudSim in cloud computing research.
2. Mention different types of schedulers in CloudSim.
3. Describe the difference between VM allocation and scheduling.
4. Identify components in a CloudSim simulation.
5. Explain the structure of a data center in CloudSim.

POST VIVA QUESTIONS

1. Explain how your custom algorithm improves over existing ones.
2. Suggest cases where Round-Robin may fail.
3. Share simulation results and observed performance.
4. Describe how cloudlets were assigned.
5. Propose future enhancements for your scheduler.

Experiment 6: Transfer Files from One Virtual Machine to Another

AIM

To transfer files between two virtual machines within the same or different networks.

SOFTWARE REQUIRED

- Two Virtual Machines (e.g., VirtualBox or VMware)
- Ubuntu or CentOS
- OpenSSH or SCP utility

BACKGROUND THEORY

Transferring files between VMs is a common need in cloud environments. Tools like `scp`, `rsync`, or shared folders can be used. SSH must be enabled on the target VM.

ALGORITHM (Using SCP)

1. Ensure both VMs are connected (via bridged/NAT network).
2. Enable SSH on the destination VM.
3. Use the `scp` command from the source VM:
4. `scp /path/to/file user@destination_ip:/path/to/target/`

CODE / COMMAND

```
# On Source VM
scp /home/user1/report.txt user2@192.168.1.15:/home/user2/
```

SAMPLE OUTPUT

```
report.txt          100% 10KB 2.0MB/s 00:00
```

RESULT

File was successfully transferred between virtual machines using the SCP protocol over SSH.

PRE VIVA QUESTIONS

1. State the need for file transfer between VMs.
2. Mention protocols used for secure file transfer.
3. Differentiate between shared folders and scp.
4. Explain the role of IP configuration in VM communication.
5. List tools used for file transfer in Linux.

POST VIVA QUESTIONS

1. Describe any issue faced during the file transfer.
2. Suggest alternative methods for large file transfers.
3. Share a real-world use of VM-to-VM transfer.
4. Describe how to make transfers permanent or automated.
5. Suggest improvements to your process.

Experiment 7: Launch a Virtual Machine Using TryStack (OpenStack Demo)

AIM

To launch a virtual machine instance using the online OpenStack demo platform **TryStack**.

SOFTWARE REQUIRED

- Browser
- Internet access
- OpenStack TryStack credentials

BACKGROUND THEORY

TryStack is an OpenStack-powered sandbox environment that allows users to try OpenStack without setup. It provides an easy GUI to launch virtual machines.

ALGORITHM

1. Login to <https://trystack.org>
2. Navigate to **Instances > Launch Instance**
3. Choose image (e.g., Ubuntu)
4. Configure flavor, security group, and key pair
5. Launch and access the instance

LONG STEPS

Step 1: Open TryStack

Step 2: Select Horizon > Compute > Instances

Step 3: Click “Launch Instance”

Step 4: Fill form:

Name: MyVM

Image: Ubuntu 20.04

Flavor: m1.small

Keypair: TryStackKey

Network: public

Step 5: Click Launch

SAMPLE OUTPUT

Instance "MyVM" - ACTIVE

IP: 172.24.4.5

Status: Running

RESULT

A virtual machine instance was successfully launched on TryStack using OpenStack Horizon Dashboard.

PRE VIVA QUESTIONS

1. Explain the purpose of using TryStack.
2. State the function of a keypair in OpenStack.
3. Mention types of OpenStack services.
4. Differentiate between flavor and image.
5. Define an instance in OpenStack.

POST VIVA QUESTIONS

1. Describe how the instance was configured.
2. Explain how to connect to the instance.
3. Share the role of networking in launching VMs.
4. Suggest improvements to OpenStack UI.
5. Mention security concerns in cloud VM creation.

Experiment 8: Install Hadoop Single Node Cluster and Run Word Count

AIM

To install Hadoop on a single-node cluster and run a basic WordCount MapReduce application.

SOFTWARE REQUIRED

- Ubuntu Linux
- Java JDK 8
- Hadoop 3.x
- Terminal / Bash

BACKGROUND THEORY

Hadoop is a framework for distributed storage and processing using MapReduce. Single-node cluster setup allows standalone testing of HDFS and MapReduce.

ALGORITHM

1. Install Java
2. Download and configure Hadoop
3. Format HDFS namenode
4. Start Hadoop services
5. Run WordCount using input files in HDFS

LONG CODE / COMMANDS

Install Java

```
sudo apt install openjdk-8-jdk
```

Download Hadoop

```
wget https://downloads.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz
```

```
tar -xvzf hadoop-3.3.6.tar.gz
```

```
mv hadoop-3.3.6 hadoop
```

```
# Configure environment
nano ~/.bashrc

# Add:
export HADOOP_HOME=~/.hadoop
export PATH=$PATH:$HADOOP_HOME/bin

# Format namenode
hdfs namenode -format

# Start Hadoop (Standalone)
start-dfs.sh
start-yarn.sh

# Create input/output dirs
hdfs dfs -mkdir /input
hdfs dfs -put sample.txt /input

# Run Word Count
hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar
wordcount /input /output

# Get result
hdfs dfs -cat /output/part-r-00000
```

SAMPLE OUTPUT

```
Hadoop 3
MapReduce 2
Word 5
```

RESULT

A Hadoop single-node cluster was successfully configured, and the Word Count application ran and produced expected output using HDFS and MapReduce.

PRE VIVA QUESTIONS

1. State the purpose of HDFS in Hadoop.
2. Describe the role of NameNode and DataNode.
3. Mention the importance of Java in Hadoop.
4. Identify the output of Map and Reduce stages.
5. List directories used by Hadoop by default.

POST VIVA QUESTIONS

1. Explain the process of uploading input to HDFS.
2. Interpret the output of Word Count.
3. Share issues faced during Hadoop installation.
4. Suggest performance tuning ideas.
5. Describe the difference between standalone and cluster mode.

INNOVATIVE APPROACHES TO THE EXPERIMENT

1. AI-Powered Chatbot Web App on GAE (Python)

Innovation: Integrate a simple chatbot using rule-based logic or open-source AI models in Python (Flask), hosted on GAE.

Features:

- User input through a web form
- Respond with predefined or dynamically generated answers
- Can be extended to use NLP libraries like NLTK or spaCy

Use Case:

- Educational bots for campus queries, FAQs, or mini health assistants.

2. Real-Time Feedback Collector with Firestore Backend (Python or Java)

Innovation: Extend the Flask or Java app to collect user feedback and store it in Google Cloud Firestore.

Features:

- Feedback form
- Store responses in Firestore database
- Admin dashboard to view entries (optional)

Use Case:

- Collect feedback from students, faculty, or conference attendees in real time.

3. Weather Forecast App using REST API Integration

Innovation: Consume third-party weather APIs (like OpenWeatherMap) in your web app and deploy it on GAE.

Features:

- Input: City name
- Output: Current temperature, humidity, condition

- Use: `requests` in Python or `HttpURLConnection` in Java

Use Case:

- Embed into campus dashboards or geography-related courses.

4. Resume Parser / Analyzer App (Python with NLP)

Innovation: A web app where users upload their resume, and the system extracts key details (skills, education, etc.) using NLP and displays analysis.

Features:

- File upload
- Text extraction using `pdfminer` or `docx`
- NLP-based parsing and keyword extraction
- Hosted on GAE

Use Case:

- Placement offices or career support centers.

5. COVID-19 Safety Guidelines Checker (Multi-language Support)

Innovation: A simple app that displays current COVID-19 safety guidelines and supports multilingual output using translation APIs.

Features:

- Select language dropdown
- Dynamic translation via Google Translate API
- GAE handles the backend and static hosting

Use Case:

- Public health awareness in multilingual regions.

6. AI-Based Text Sentiment Analyzer (Python)

Innovation: A GAE-deployed Flask app where users enter a sentence and get a sentiment score (positive/negative/neutral) using NLP.

Features:

- Input box
- Backend using TextBlob or VADER
- Output sentiment classification

Use Case:

- Educational demo of AI or integration into feedback systems.

7. Expense Tracker App with Google Sheets API Integration

Innovation: Web app where users enter their daily expenses and the backend stores it in Google Sheets via Sheets API.

Features:

- Expense form
- Authentication via OAuth (optional)
- Updates cloud-based sheet in real time

Use Case:

- Budget tracking for students, clubs, or teams.

8. Python + Java Hybrid App via REST API

Innovation: Deploy a Python Flask app on GAE that communicates with a Java-based backend microservice via HTTP.





Features:

- Separation of UI (Python) and logic (Java)
- Demonstrates service-to-service communication on cloud

Use Case:

- Introduces students to microservice architecture on GCP.

Why These Are Innovative:

-  **Integration** of APIs and cloud services
-  **Application** of AI/ML/NLP in web apps
-  Use of **real-time databases** and authentication
-  Enhancing user interactivity and analytics