

ProgressSoft Assignment Solutions:

Mohamed Mohamed Shehata Badawy

Backend Developer & Cloud DevOps Engineer

Github Repository



<https://github.com/shehata18/ProgressSoft-Assignment>

ProgressSoft Assignment Solutions:

Section 1: Linux

1. OS Information Script

To gather comprehensive system information and resource utilization details, a Bash script named `system_info.sh` was developed. This script leverages standard Linux commands to extract data such as the executing user, hostname, IP addresses, operating system type and version, kernel version, architecture, virtualization technology, server time, time zone, system uptime, total memory, memory usage, swap usage, and the number of CPU cores. The script is designed to provide a quick overview of the system's configuration and current state.

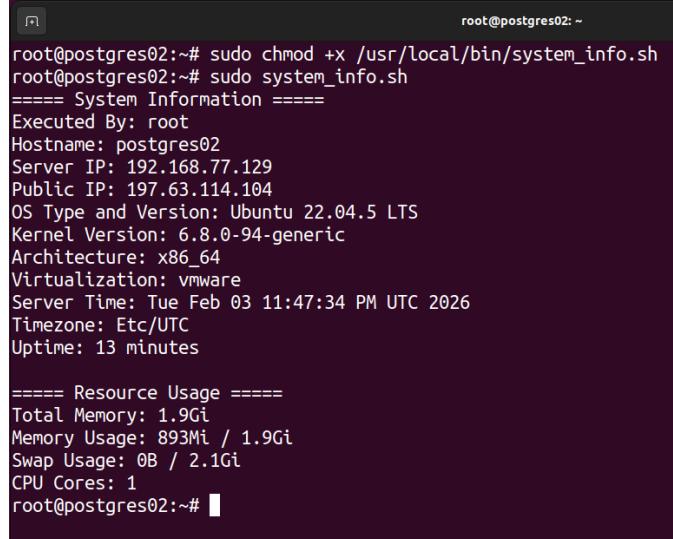
Script Content (`system_info.sh`):

```
#!/bin/bash

export LANG=en_US.UTF-8
export LC_ALL=en_US.UTF-8

echo "===== System Information ====="
echo "Executed By: $(whoami)"
echo "Hostname: $(hostname)"
echo "Server IP: $(hostname -I | awk '{print $1}')"
echo "Public IP: $(curl -s ifconfig.me)"
echo "OS Type and Version: $(lsb_release -d | cut -f2)"
echo "Kernel Version: $(uname -r)"
echo "Architecture: $(uname -m)"
echo "Virtualization: $(systemctl-detect-virt)"
echo "Server Time: $(LC_TIME=en_US.UTF-8 date '+%a %b %d %I:%M:%S %p %Z %Y')"
echo "Timezone: $(timedatectl show -p Timezone --value)"
echo "Uptime: $(awk '{print int($1/60) \" minutes\"}' /proc/uptime)"

echo "===== Resource Usage ====="
echo "Total Memory: $(free -h | awk '/Mem:/ {print $2})"
echo "Memory Usage: $(free -h | awk '/Mem:/ {print $3 \" / \" $2})"
echo "Swap Usage: $(free -h | awk '/Swap:/ {print $3 \" / \" $2})"
echo "CPU Cores: $(nproc)"
```



The terminal window shows the command `sudo chmod +x /usr/local/bin/system_info.sh` being run, followed by the execution of the script with `sudo system_info.sh`. The output displays system information including the executing user (root), hostname (postgres02), server IP (192.168.77.129), public IP (197.63.114.104), OS type and version (Ubuntu 22.04.5 LTS), kernel version (6.8.0-94-generic), architecture (x86_64), virtualization (vmware), server time (Tue Feb 03 11:47:34 PM UTC 2026), timezone (Etc/UTC), and uptime (13 minutes). It also shows resource usage statistics like total memory (1.9Gi), memory usage (893Mi / 1.9Gi), swap usage (0B / 2.1Gi), and CPU cores (1).

2. User and Group Management:

To establish a new user account with specific group memberships, the following commands were utilized. A user named PS was created, assigned to the primary group PSgroup , and also added to the secondary group dba . This setup ensures appropriate permissions and access control for the user within the system.

```

```
sudo groupadd PSgroup
```

```
sudo groupadd dba
```

```
sudo useradd -m -g PSgroup -G dba PS
```

```
sudo passwd PS
```

```
id PS
```

```

3. Root Password Modification:

```

```
sudo passwd root
```

```

```
root@postgres02:~#  
root@postgres02:~# sudo groupadd PSgroup  
root@postgres02:~# sudo groupadd dba  
root@postgres02:~# sudo useradd -m -g PSgroup -G dba PS  
root@postgres02:~# sudo passwd PS  
New password:  
BAD PASSWORD: The password is shorter than 8 characters  
Retype new password:  
passwd: password updated successfully  
root@postgres02:~#  
root@postgres02:~#  
root@postgres02:~# passwd root  
New password:  
Retype new password:  
passwd: password updated successfully  
root@postgres02:~#
```

4. Software Installation (MySQL & HAProxy)

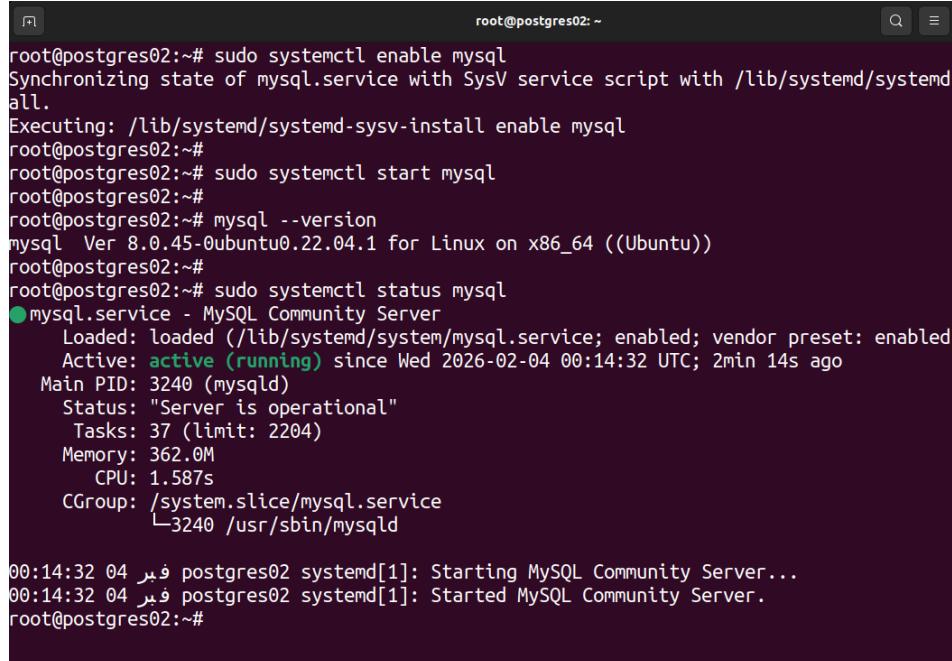
MySQL Server and HAProxy were installed on the Ubuntu system using the `apt install` command. MySQL provides robust relational database capabilities, while HAProxy acts as a high-performance load balancer and reverse proxy. Verifying their installed versions confirms successful deployment.

Install MySQL

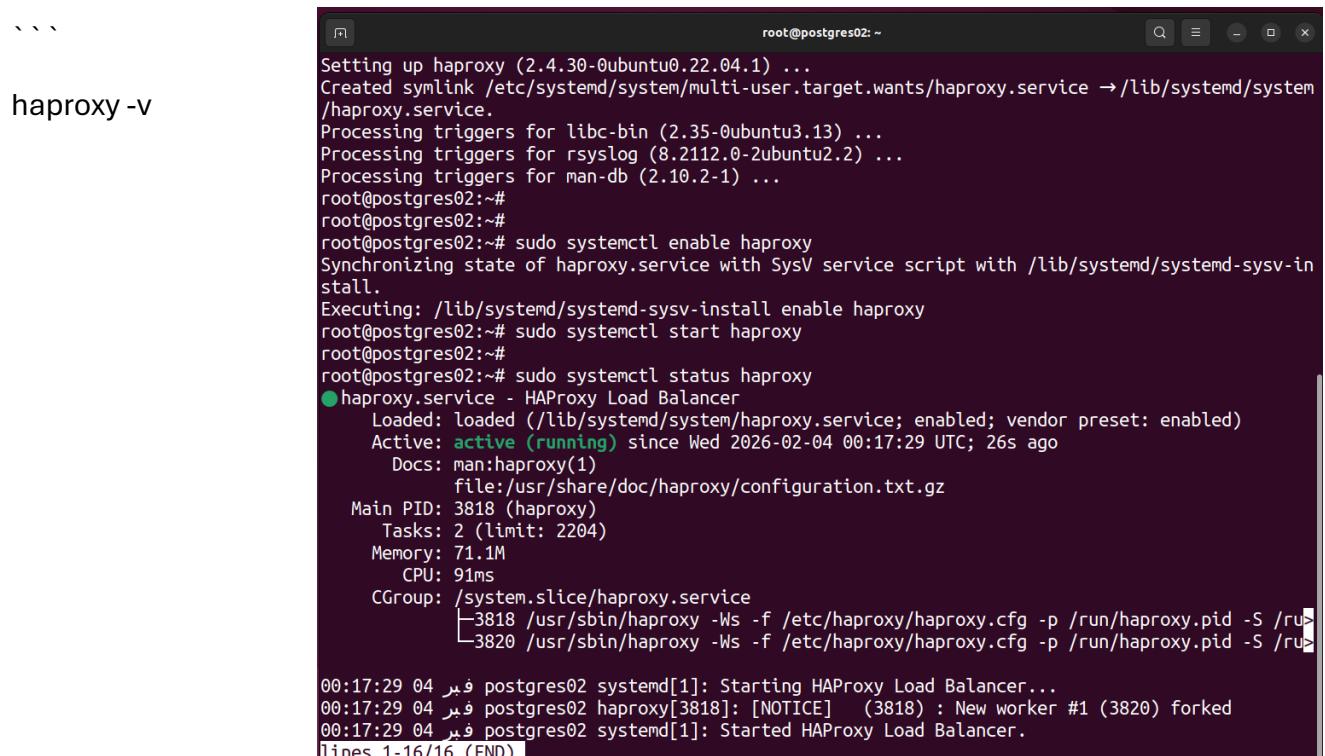
```
sudo apt install mysql-server -y  
sudo systemctl enable mysql  
sudo systemctl start mysql  
mysql --version
```

Install HAProxy

```
sudo apt install haproxy -y  
sudo systemctl enable haproxy  
sudo systemctl start haproxy
```



```
sudo apt install haproxy -y  
sudo systemctl enable haproxy  
sudo systemctl start haproxy
```



5. Allow TCP/UDP only on port 3306 (Firewall)

Step 1: Enable UFW

```
```
sudo ufw enable
````
```

Step 2: Allow MySQL Port (TCP & UDP)

```
```
sudo ufw allow 3306/tcp
sudo ufw allow 3306/udp
````
```

Step 3: Deny everything else

```
```
sudo ufw default deny incoming
sudo ufw default allow outgoing
````
```

Check status => sudo ufw status

```
root@postgres02:~# sudo ufw enable
Firewall is active and enabled on system startup
root@postgres02:#
root@postgres02:~# sudo ufw allow 3306/tcp
Rule added
Rule added (v6)
root@postgres02:~# sudo ufw allow 3306/udp
Rule added
Rule added (v6)
root@postgres02:#
root@postgres02:~# echo "For Deny Everything else"
For Deny Everything else
root@postgres02:#
root@postgres02:~# sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
root@postgres02:#
root@postgres02:~# sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
root@postgres02:#
root@postgres02:~# sudo ufw status
Status: active

To                         Action      From
--                         --          --
3306/tcp                   ALLOW       Anywhere
3306/udp                   ALLOW       Anywhere
3306/tcp (v6)              ALLOW       Anywhere (v6)
3306/udp (v6)              ALLOW       Anywhere (v6)

root@postgres02:~#
```

6. File Transfer Simulation

To demonstrate file transfer capabilities, a simulation was performed where a file from the local machine is copied to the virtual machine. While various methods exist, **scp** (Secure Copy Protocol) is a common command-line tool for secure file transfers over SSH.

From windows terminal or powershell

```

**scp file.txt shehata@192.168.77.129:/home/shehata/**

**sftp shehata@192.168.77.129**

**put file.txt**

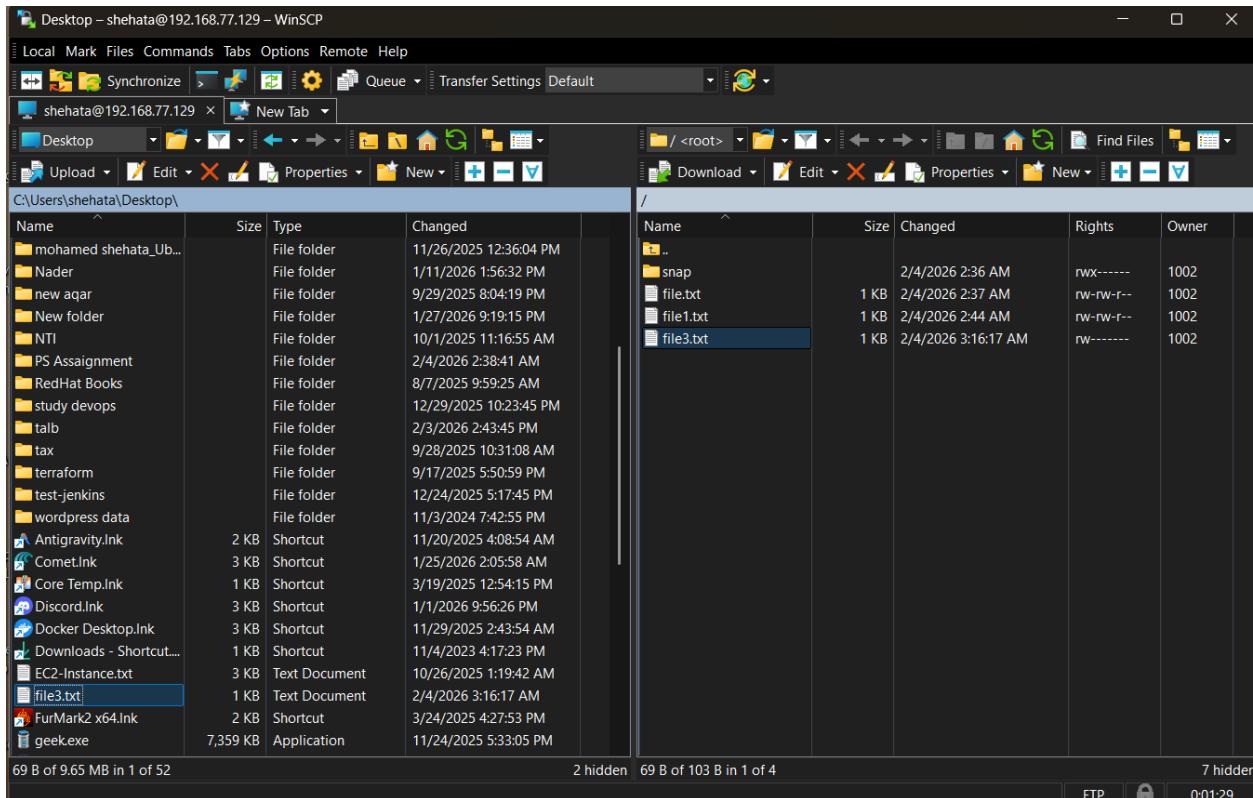
```

```
█ ~\Desktop ➤ scp file.txt shehata@192.168.77.129:/home/shehata/
shehata@192.168.77.129's password:
file.txt                                              100%   12      3.9KB/s  00:00
█ ~\Desktop
█ ~\Desktop
█ ~\Desktop ➤ sftp shehata@192.168.77.129
shehata@192.168.77.129's password:
Connected to 192.168.77.129.
sftp> put file1.txt
Uploading file1.txt to /home/shehata/file1.txt
file1.txt                                              100%   22      10.7KB/s  00:00
sftp> exit
```

```
CPU: 18ms
CGroup: /system.slice/ssh.service
└─5938 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

00:36:08 04 فبر postres02 systemd[1]: Starting OpenBSD Secure Shell server...
00:36:08 04 فبر postres02 sshd[5938]: Server listening on 0.0.0.0 port 22.
00:36:08 04 فبر postres02 sshd[5938]: Server listening on :: port 22.
00:36:08 04 فبر postres02 systemd[1]: Started OpenBSD Secure Shell server.
shehata@postgres02: $ ^C
shehata@postgres02: $ sudo systemctl start ssh
sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
shehata@postgres02: $ sudo ufw allow ssh
sudo ufw reload
Rule added
Rule added (v6)
Firewall reloaded
shehata@postgres02: $ ls
file.txt  snap
shehata@postgres02: $ cat file.txt
Hello World!shehata@postgres02: $
shehata@postgres02: $
shehata@postgres02: $ ls
file1.txt  file.txt  snap
shehata@postgres02: $ cat file1.txt
Hello from SFTP methodshehata@postgres02: $
shehata@postgres02: $
shehata@postgres02: $ ls
file1.txt  file3.txt  file.txt  snap
shehata@postgres02: $ cat file3.txt
-----
Hello from ftp method
-----shehata@postgres02: $ S
```

We can use WinSCP GUI at windows and use FTP



7. Read about Linux in general (architecture, directories, distributions, etc.).

- small summary:

Linux is a free, open-source operating system based on Unix. It is mainly used in servers, cloud computing, DevOps, and embedded systems.

Linux has a layered architecture: hardware at the bottom, the kernel in the core (managing processes, memory, files, devices, and networking), and user space on top where applications and users interact with the system.

Linux uses a single filesystem hierarchy starting from the root directory `/`. Important directories include `/etc` for configuration, `/home` for users, `/var` for logs and variable data, `/bin` for commands, and `/proc` for process and kernel information.

A Linux distribution (distro) is the Linux kernel combined with system tools and a package manager. Popular distros include **Ubuntu**, **Debian**, **CentOS**, **RHEL**, and **Arch Linux**.

Linux is multi-user, secure, stable, and highly customizable, which makes it the most widely used operating system for servers and cloud environments.

Section 2: SQL

Q: Read about SQL/PLSQL and DML, DDL, and DCL?

SQL (Structured Query Language):

- SQL is a standard language for managing and manipulating relational databases
- Used to query, insert, update, and delete data

PL/SQL (Procedural Language/SQL):

- Oracle's procedural extension to SQL
- Allows you to write programs with variables, conditions, loops, and exception handling
- Combines SQL's data manipulation with procedural programming features

DML (Data Manipulation Language):

- Commands used to manipulate data in database tables
- Examples: SELECT, INSERT, UPDATE, DELETE
- Used for querying and modifying data

DDL (Data Definition Language):

- Commands used to define database structure/schema
- Examples: CREATE, ALTER, DROP, TRUNCATE
- Used for creating and modifying database objects

DCL (Data Control Language):

- Commands used to control access to data
- Examples: GRANT, REVOKE
- Used for managing permissions and security

Q: What are the different types of JOINS?

1. INNER JOIN:

- Returns only matching rows from both tables
- Most common type of join

2. LEFT JOIN (LEFT OUTER JOIN):

- Returns all rows from left table and matching rows from right table
- NULL values for non-matching rows from right table

3. RIGHT JOIN (RIGHT OUTER JOIN):

- Returns all rows from right table and matching rows from left table
- NULL values for non-matching rows from left table

4. FULL OUTER JOIN:

- Returns all rows from both tables
- NULL values where there's no match

5. CROSS JOIN:

- Returns Cartesian product of both tables
- Every row from first table combined with every row from second table

6. SELF JOIN:

- Table joined with itself
- Useful for hierarchical data

Q: What is RDBMS and NoSQL?

RDBMS (Relational Database Management System):

- Stores data in structured tables with rows and columns
- Uses SQL for querying
- ACID compliant (Atomicity, Consistency, Isolation, Durability)
- Examples: Oracle, MySQL, PostgreSQL, SQL Server
- Best for: Structured data, complex queries, transactions

NoSQL (Not Only SQL):

- Stores data in non-tabular format
- Types: Document (MongoDB), Key-Value (Redis), Column-family (Cassandra), Graph (Neo4j)
- Flexible schema
- Horizontally scalable
- Best for: Unstructured data, high volume, rapid development

Aggregation Functions:

- **COUNT()**: Counts number of rows
- **SUM()**: Calculates sum of numeric column
- **AVG()**: Calculates average value
- **MAX()**: Returns maximum value
- **MIN()**: Returns minimum value
- **GROUP BY**: Groups rows sharing a property

Date and String Functions:

Date Functions: SYSDATE, TO_DATE, TO_CHAR, ADD_MONTHS, MONTHS_BETWEEN, TRUNC, EXTRACT

String Functions: CONCAT, SUBSTR, LENGTH, UPPER, LOWER, TRIM, REPLACE, INSTR

Constraints:

- PRIMARY KEY: Uniquely identifies each row
- FOREIGN KEY: Links tables together
- NOT NULL: Column cannot have NULL values
- UNIQUE: All values must be unique
- CHECK: Validates values meet specific condition
- DEFAULT: Sets default value

Indexes:

- Database objects that improve query performance
- Created on columns frequently used in WHERE clauses
- Trade-off: Faster reads, slower writes
- Types: B-tree (default), Bitmap, Unique, Composite

Q1: Create Tables with Constraints

```
1  -- Create MyDepartment table
2  CREATE TABLE MyDepartment (
3      Dept_ID NUMBER PRIMARY KEY,
4      Name VARCHAR2(100) NOT NULL
5  );
6
7  -- Create Gender table
8  CREATE TABLE Gender (
9      Gender_ID NUMBER PRIMARY KEY,
10     Name VARCHAR2(20) NOT NULL,
11     CONSTRAINT chk_gender_name CHECK (Name IN ('Male', 'Female'))
12 );
13
14 -- Create University table
15 CREATE TABLE University (
16     ID NUMBER PRIMARY KEY,
17     Name VARCHAR2(200) NOT NULL
18 );
```

```
1  -- Create MyEmployee table
2  CREATE TABLE MyEmployee (
3      ID NUMBER PRIMARY KEY,
4      LAST_NAME VARCHAR2(50) NOT NULL,
5      FIRST_NAME VARCHAR2(50) NOT NULL,
6      HIRE_DATE DATE NOT NULL,
7      USERID NUMBER UNIQUE NOT NULL,
8      SALARY NUMBER NOT NULL,
9      DEPT_ID NUMBER,
10     Gender_ID NUMBER NOT NULL,
11     University_ID NUMBER,
12     EMP_IMAGE BLOB,
13     MANAGER_ID NUMBER,
14     JOB_TITLE VARCHAR2(50),
15     CONSTRAINT fk_dept FOREIGN KEY (DEPT_ID) REFERENCES MyDepartment(Dept_ID),
16     CONSTRAINT fk_gender FOREIGN KEY (Gender_ID) REFERENCES Gender(Gender_ID),
17     CONSTRAINT fk_university FOREIGN KEY (University_ID) REFERENCES University(ID),
18     CONSTRAINT fk_manager FOREIGN KEY (MANAGER_ID) REFERENCES MyEmployee(ID),
19     CONSTRAINT chk_salary CHECK (SALARY > 0)
20 );
```

Note: Added for manager relationship , added job_title for Q3

Q2: Write a SQL query to retrieve Employee Data with joins

```
1  SELECT
2      e.FIRST_NAME || ' ' || e.LAST_NAME AS "Employee Name",
3      e.SALARY AS "Salary",
4      d.Name AS "Department Name",
5      m.FIRST_NAME || ' ' || m.LAST_NAME AS "Manager",
6      g.Name AS "Gender",
7      u.Name AS "Employee University"
8  FROM
9      MyEmployee e
10     LEFT JOIN
11          MyDepartment d ON e.DEPT_ID = d.Dept_ID
12     LEFT JOIN
13          MyEmployee m ON e.MANAGER_ID = m.ID
14     INNER JOIN
15          Gender g ON e.Gender_ID = g.Gender_ID
16     LEFT JOIN
17          University u ON e.University_ID = u.ID;
```

The screenshot shows the FreeSQL interface with the following details:

- Toolbar:** Includes icons for file operations (New, Open, Save, Print, Copy, Paste, Find, Replace, Undo, Redo), a zoom slider, and font size controls.
- Header:** Shows the title "[SQL Worksheet]*", the user "26ai", a "Connect to the Database" button, a "Help and Feedback" link, and a "Sign Out" link.
- Navigator:** Displays the schema structure with tables: GENDER, MYDEPARTMENT, MYEMPLOYEE, and UNIVERSITY.
- Code Editor:** Contains the SQL query provided in the question, with syntax highlighting for keywords and identifiers.
- Query Result:** A table showing the results of the query execution. The table has 7 columns: EMPLOYEE NAME, SALARY, DEPARTMENT NAME, MANAGER, GENDER, and EMPLOYEE UNIVERSITY. The data is as follows:

| | EMPLOYEE NAME | SALARY | DEPARTMENT NAME | MANAGER | GENDER | EMPLOYEE UNIVERSITY |
|---|---------------|--------|-----------------|---------------|--------|----------------------|
| 1 | SARA MALEK | 3500 | Sales | AHMED MOHAMED | Female | Harvard University |
| 2 | ALI HISHAM | 4500 | IT | AHMED MOHAMED | Male | MIT |
| 3 | HASSAN HAKEEM | 3200 | Sales | AHMED MOHAMED | Male | MIT |
| 4 | SALY SHAHEEN | 3800 | HR | AHMED MOHAMED | Male | Stanford University |
| 5 | RAMI KHALIL | 4200 | IT | ALI HISHAM | Male | University of Jordan |
| 6 | AHMED MOHAMED | 5000 | IT | | Male | Harvard University |
| 7 | MAHMOUD KAMAL | 6000 | Finance | | Female | Stanford University |

Q3: Display all job titles and the total monthly salary for each job title, where the total payroll exceeds \$2500, excluding sales employees.

```
1 SELECT
2     JOB_TITLE,
3         SUM(SALARY) AS "Total Monthly Salary"
4 FROM
5     MyEmployee
6 WHERE
7     UPPER(JOB_TITLE) NOT LIKE '%SALES%'
8 GROUP BY
9     JOB_TITLE
10 HAVING
11     SUM(SALARY) > 2500
12 ORDER BY
13     "Total Monthly Salary" DESC;
```

The screenshot shows the FreeSQL application interface. The top navigation bar includes 'FreeSQL', 'Worksheet', 'Library', '26ai' (dropdown), 'Connect to the Database', 'Help and Feedback', and a user icon. On the left, there's a 'Navigator' panel with 'My Schema' dropdown, 'Tables' dropdown, and a search bar. Below these are lists for 'GENDER', 'MYDEPARTMENT', 'MYEMPLOYEE', and 'UNIVERSITY'. The main workspace is titled '[SQL Worksheet]*' and contains the SQL query from the previous code block. Below the query is a 'Query result' table with four rows. The table has two columns: 'JOB_TITLE' and 'TOTAL MONTHLY SALARY'. The data is as follows:

| | JOB_TITLE | TOTAL MONTHLY SALARY |
|---|-----------------|----------------------|
| 1 | Developer | 8700 |
| 2 | Finance Manager | 6000 |
| 3 | Manager | 5000 |
| 4 | HR Specialist | 3800 |

Q4: There are four coding errors in the following statement. Identify them:

```
SELECT empno, ename,  
salary x 12 ANNUAL SALARY; FROM emp;
```

Errors identified

1. Missing multiplication operator (*)

- "salary x 12" should be "salary * 12"

2. Column alias needs quotes or underscore

- "ANNUAL SALARY" should be "ANNUAL_SALARY" or "ANNUAL SALARY" (in quotes)

3. Semicolon in wrong position

- Semicolon should be at the end, not before FROM

4. Incorrect statement structure

- FROM clause should come before the semicolon

Correct statement structure

```
1  SELECT  
2      empno,  
3      ename,  
4      salary * 12 AS ANNUAL_SALARY  
5  FROM  
6      emp;  
7  
8  -- Alternative with quoted alias:  
9  SELECT  
10     empno,  
11     ename,  
12     salary * 12 AS "ANNUAL SALARY"  
13  FROM  
14     emp;
```

Q5: Oracle function – F_HR_QUERY



```
1 CREATE OR REPLACE FUNCTION F_HR_QUERY
2 RETURN SYS_REFCURSOR
3 IS
4     v_scott_hire_date DATE;
5     result_cursor SYS_REFCURSOR;
6 BEGIN
7     -- Get SCOTT's hire date
8     SELECT HIRE_DATE
9         INTO v_scott_hire_date
10        FROM MyEmployee
11       WHERE UPPER(LAST_NAME) = 'SCOTT';
12
13     -- Open cursor with employees hired after SCOTT
14     OPEN result_cursor FOR
15         SELECT
16             FIRST_NAME || ' ' || LAST_NAME AS Employee_Name,
17             HIRE_DATE
18         FROM
19             MyEmployee
20         WHERE
21             HIRE_DATE > v_scott_hire_date
22         ORDER BY
23             HIRE_DATE;
24
25     RETURN result_cursor;
26
27 EXCEPTION
28     WHEN NO_DATA_FOUND THEN
29         DBMS_OUTPUT.PUT_LINE('Employee SCOTT not found');
30         RETURN NULL;
31     WHEN OTHERS THEN
32         DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
33         RETURN NULL;
34 END F_HR_QUERY;
35 /
```

Q6: Oracle function – P_COPY_EMPLOYEE

Statement details

```
CREATE OR REPLACE PROCEDURE P_COPY_EMPLOYEE
IS
    v_count NUMBER;
BEGIN
    -- Clear existing data in target table
    DELETE FROM MyEmployee_update;

    -- Copy all data from MyEmployee to MyEmployee_update
    INSERT INTO MyEmployee_update
    SELECT * FROM MyEmployee;

    -- Get count of inserted records
    v_count := SQL%ROWCOUNT;

    -- Commit the transaction
    COMMIT;

    -- Display success message
    DBMS_OUTPUT.PUT_LINE('Successfully copied ' || v_count ||
        ' records from MyEmployee to MyEmployee_update');

EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('Error occurred: ' || SQLERRM);
        RAISE;
END P_COPY_EMPLOYEE;
```

 Insert Into Editor

Execute:

EXEC P_COPY_EMPLOYEE

Result

Procedure P_COPY_EMPLOYEE compiled
Elapsed: 00:00:00.044




[SQL Worksheet]* ▾ ▶ ⏪ ⏩ ⏴ ⏵ ⏳ Aa └ ┌

1 `SELECT COUNT(*) FROM MyEmployee_update`

Navigator **Files**

My Schema

Tables

Search objects

GENDER
MYDEPARTMENT
MYEMPLOYEE
MYEMPLOYEE_UPDATE
UNIVERSITY

Query result Script output DBMS output Explain Plan SQL history

  Download ▾ Execution time: 0.003 seconds

| | COUNT(*) |
|---|----------|
| 1 | 7 |

Tomcat Section:

1. In your own words, explain what JVM means ?

The **JVM (Java Virtual Machine)** is a runtime environment that executes Java bytecode.

It allows Java applications to run on any operating system without modification by abstracting the underlying hardware and OS.

Responsibilities of JVM:

- Loads and verifies bytecode
- Manages memory (Heap, Stack, Garbage Collection)
- Executes Java instructions
- Provides platform independence (“Write once, run anywhere”)

What is an Application Server?

An **application server** is software that provides an environment to run backend applications and manage:

- Request handling
- Session management
- Security
- Thread management
- Resource pooling (DB connections)

Examples:

Apache Tomcat (Servlet container), JBoss / WildFly, WebLogic, WebSphere

Tomcat is a lightweight application server, mainly for Java web apps (Servlets & JSP).

Q3: What is a WAR file? How Tomcat handles it?

⇒ A **WAR (Web Application Archive)** file is a packaged Java web application.

It contains:

- Compiled Java classes
- JSP files
- Configuration files (web.xml)
- Libraries (WEB-INF/lib)

How Tomcat handles a WAR?

- When a WAR file is placed in Tomcat's webapps/ directory
- Tomcat automatically extracts it
- Deploys it as a web application

Deployment location: `$TOMCAT_HOME/webapps/`

Practice:

1. Download and install Tomcat as a WebApp container. Set up a reverse proxy with Nginx and connect both so that the WebApp can be accessed through a DNS name over port 80.

...

`sudo apt update`

`sudo apt install openjdk-11-jdk -y`

...

➤ Verify : `java -version`

➤ Download & Install Tomcat 9

```

```
cd /opt
```

```
sudo wget
```

```
https://archive.apache.org/dist/tomcat/tomcat9/v9.0.89/bin/apache-tomcat-9.0.89.tar.gz
```

```
sudo tar -xvf apache-tomcat-9.0.89.tar.gz
```

```
sudo mv apache-tomcat-9.0.89 tomcat9
```

```

➤ Start Tomcat:

```
/opt/tomcat9/bin/startup.sh
```

➤ Access:

```
http://VM_IP:8080
```

The screenshot shows a terminal window with the following session:

```
shehata@postgres02:/opt/tomcat9$ ls
bin BUILDING.txt conf CONTRIBUTING.md lib LICENSE logs NOTICE README.md RELEASE-NOTES RUNNING.txt temp webapps work
shehata@postgres02:/opt/tomcat9$ sudo -i
root@postgres02:~# ls
snap
root@postgres02:~# cd /opt/
root@postgres02:/opt# ls
apache-tomcat-9.0.89.tar.gz tomcat9
root@postgres02:/opt# cd tomcat9/
root@postgres02:/opt/tomcat9# ls
bin BUILDING.txt conf CONTRIBUTING.md lib LICENSE logs NOTICE README.md RELEASE-NOTES RUNNING.txt temp webapps work
root@postgres02:/opt/tomcat9# cd bin
root@postgres02:/opt/tomcat9/bin# ls
bootstrap.jar ciphers.sh daemon.sh setclasspath.bat startup.sh version.bat
catalina.bat commons-daemon.jar digest.bat setclasspath.sh tomcat-juli.jar version.sh
catalina.sh commons-daemon-native.tar.gz digest.sh shutdown.bat tomcat-native.tar.gz
catalina-tasks.xml configtest.bat makebase.bat shutdown.sh tool-wrapper.bat
ciphers.bat configtest.sh makebase.sh startup.sh tool-wrapper.sh
root@postgres02:/opt/tomcat9/bin# ./startup.sh
-bash: ./startup.sh: No such file or directory
root@postgres02:/opt/tomcat9/bin# ./startup.sh
Using CATALINA_BASE: /opt/tomcat9
Using CATALINA_HOME: /opt/tomcat9
Using CATALINA_TMPDIR: /opt/tomcat9/temp
Using JRE_HOME: /usr
Using CLASSPATH: /opt/tomcat9/bin/bootstrap.jar:/opt/tomcat9/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
root@postgres02:/opt/tomcat9/bin#
```

The terminal window shows the contents of the `server.xml` file, which includes various configuration sections like `<Host>`, `<Context>`, and `<Connector>`. The browser window displays the Apache Tomcat 9.0.89 homepage, featuring a cartoon cat logo, a success message, and links for documentation, developer quick start, and getting help.

```

</div>
</div>
<div class="col20">
<div class="container">
<h4>Miscellaneous</h4>
<ul>
<li><a href="https://tomcat.apache.org/contact.html">Contact</a></li>
<li><a href="https://tomcat.apache.org/legal.html">Legal</a></li>
<li><a href="https://www.apache.org/foundation/sponsorship.html">Sponsorship</a></li>
<li><a href="https://www.apache.org/foundation/thanks.html">Thanks</a></li>
</ul>
</div>
<div class="col20">
<div class="container">
<h4>Apache Software Foundation</h4>
<ul>
<li><a href="https://tomcat.apache.org/whoweare.html">Who We Are</a></li>
<li><a href="https://tomcat.apache.org/heritage.html">Heritage</a></li>
<li><a href="https://www.apache.org">Apache Home</a></li>
<li><a href="https://tomcat.apache.org/resources.html">Resources</a></li>
</ul>
</div>
<br class="separator" />
</div>
<p class="copyright">Copyright &copy;1999-2026 Apache Software Foundation. All Rights Reserved</p>
</div>
</body>
</html>

```

2. Change the Tomcat port to 7070.

- First need to edit `server.xml` file

```

**`sudo vim /opt/tomcat9/conf/server.xml`**

```

- Second Change this line to :

`<Connector port="7070"`

- Third restart tomcat

`/opt/tomcat9/bin/shutdown.sh`

`/opt/tomcat9/bin/startup.sh`

root@postgres02:/opt/tomcat9/bin# cd ..
root@postgres02:/opt/tomcat9# ls
bin BUILDING.txt conf CONTRIBUTING.md lib LICENSE logs NOTICE README.md RELEASE-NOTES RUNNING.txt temp webapps work
root@postgres02:/opt/tomcat9#
root@postgres02:/opt/tomcat9# cd conf
root@postgres02:/opt/tomcat9/conf# vim server.xml
root@postgres02:/opt/tomcat9/conf# cd bin
-bash: cd: bin: No such file or directory
root@postgres02:/opt/tomcat9/conf# cd ..
root@postgres02:/opt/tomcat9# cd bin
root@postgres02:/opt/tomcat9# ./shutdown.sh
Using CATALINA_BASE: /opt/tomcat9
Using CATALINA_HOME: /opt/tomcat9
Using CATALINA_TMPDIR: /opt/tomcat9/temp
Using JRE_HOME: /usr
Using CLASSPATH: /opt/tomcat9/bin/bootstrap.jar:/opt/tomcat9/bin/tomcat-juli.jar
Using CATALINA_OPTS:
NOTE: Picked up JDK JAVA_OPTIONS: --add-opens=java.base/java.lang=ALL-UNNAMED
--add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED
root@postgres02:/opt/tomcat9/bin# ./startup.sh
Using CATALINA_BASE: /opt/tomcat9
Using CATALINA_HOME: /opt/tomcat9
Using CATALINA_TMPDIR: /opt/tomcat9/temp
Using JRE_HOME: /usr
Using CLASSPATH: /opt/tomcat9/bin/bootstrap.jar:/opt/tomcat9/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
root@postgres02:/opt/tomcat9/bin# curl http://localhost:7070

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<title>Apache Tomcat/9.0.89</title>

The Apache Tomcat™ 9.0.89 Documentation

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 Recommended Reading:
[Security Considerations How-To](#)
[Manager Application How-To](#)
[Clustering/Session Replication How-To](#)

Developer Quick Start

Tomcat Setup
First Web Application
Realms & AAA
JDBC DataSources
Examples
Servlet Specifications
Tomcat Versions

Managing Tomcat
For security, access to the [manager webapp](#) is restricted. Users are defined in: [CATALINA_HOME/conf/tomcat-users.xml](#)
In Tomcat 9.0 access to the manager application is split between different users. [Read more...](#)

Documentation
[Tomcat 9.0 Documentation](#)
[Tomcat 9.0 Configuration](#)
[Tomcat Wiki](#)
Find additional important configuration information in: [CATALINA_HOME/running.txt](#)
Developers may be interested in:
[Tomcat 9.0 Bug Database](#)
[Tomcat 9.0 JavaDocs](#)
[Tomcat 9.0 Git Repository at GitHub](#)

Getting Help
[FAQ and Mailing Lists](#)
The following mailing lists are available:
[tomcat-announce](#)
Important announcements, releases, security vulnerability notifications. (Low volume).
[tomcat-users](#)
User support and discussion
[taglibs-user](#)
User support and discussion for Apache Taglibs
[tomcat-dev](#)
Development mailing list, including commit messages

Other Downloads
[Tomcat Connectors](#)
[Tomcat Native](#)
[Taglibs](#)
[Deployer](#)

Other Documentation
[Tomcat Connectors](#)
[mod_jk Documentation](#)
[Tomcat Native](#)
[Deployer](#)

Get Involved
[Overview](#)
[Source Repositories](#)
[Mailing Lists](#)
[Wiki](#)

Miscellaneous
[Contact](#)
[Legal](#)
[Sponsorship](#)
[Thanks](#)

Apache Software Foundation
Who We Are
Heritage
Apache Home
Resources

Nginx Reverse Proxy:

➤ Install Nginx

```
sudo apt install nginx -y
```

➤ Configure Nginx

```
sudo nano /etc/nginx/sites-available/tomcat
```

```
server {
```

```
    listen 80;
```

```
    server_name tomcat.local;
```

```
    location / {
```

```
        proxy_pass http://localhost:7070;
```

```
        proxy_set_header Host $host;
```

```
        proxy_set_header X-Real-IP $remote_addr;
```

```
    }}
```

➤ Enable page or site:

```
sudo ln -s /etc/nginx/sites-available/tomcat /etc/nginx/sites-enabled/  
sudo nginx -t  
sudo systemctl restart nginx
```

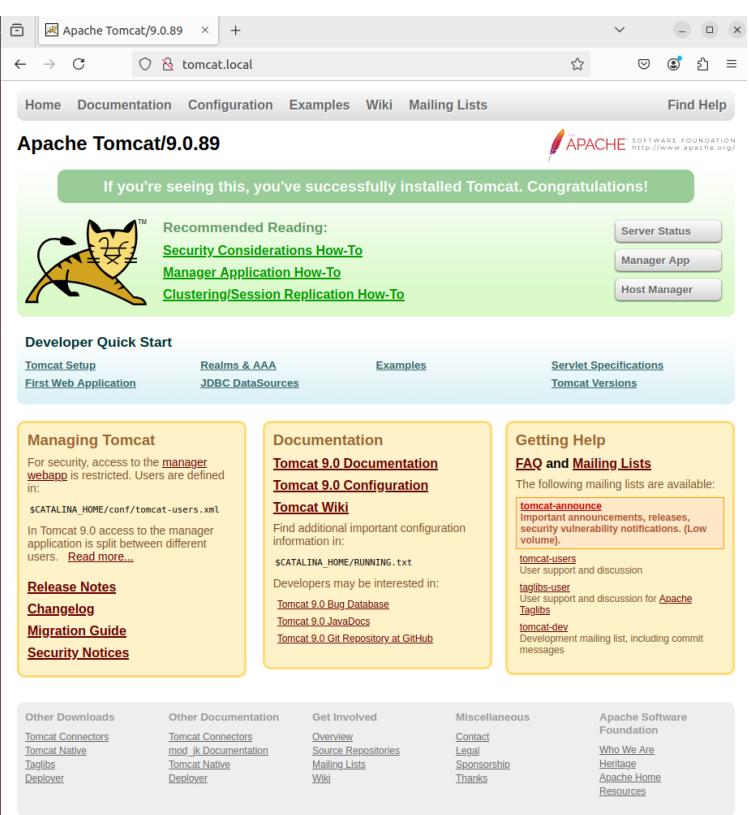
➤ To add DNS entry:

```
sudo vim /etc/hosts
```

```
127.0.0.1 tomcat.local
```

➤ Then Access it
<http://tomcat.local>

```
root@postgres02:~  
Setting up libnginx-mod-stream (1.18.0-6ubuntu14.7) ...  
Setting up libnginx-mod-stream-geoip2 (1.18.0-6ubuntu14.7) ...  
Setting up nginx-core (1.18.0-6ubuntu14.7) ...  
* Upgrading binary nginx  
Setting up nginx (1.18.0-6ubuntu14.7) ...  
Processing triggers for man-db (2.10.2-1) ...  
Processing triggers for ufw (0.36.1-4ubuntu0.1) ...  
root@postgres02:~#  
root@postgres02:~#  
root@postgres02:~# vim /etc/nginx/sites-available/tomcat  
root@postgres02:~#  
root@postgres02:~# sudo ln -s /etc/nginx/sites-available/tomcat /etc/nginx/sites-enabled/  
root@postgres02:~#  
root@postgres02:~# sudo nginx -t  
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
nginx: configuration file /etc/nginx/nginx.conf test is successful  
root@postgres02:~#  
root@postgres02:~# sudo systemctl restart nginx  
root@postgres02:~#  
root@postgres02:~# vim /etc/hosts  
root@postgres02:~#  
root@postgres02:~# curl http://tomcat.local  
  
<!DOCTYPE html>  
<html lang="en">  
  <head>  
    <meta charset="UTF-8" />  
    <title>Apache Tomcat/9.0.89</title>  
    <link href="favicon.ico" rel="icon" type="image/x-icon" />  
    <link href="tomcat.css" rel="stylesheet" type="text/css" />  
  </head>  
  
  <body>  
    <div id="wrapper">  
      <div id="navigation" class="curved container">  
        <span id="nav-home"><a href="https://tomcat.apache.org/">Home</a></span>  
        <span id="nav-hosts"><a href="/docs/">Documentation</a></span>  
        <span id="nav-config"><a href="/docs/config/">Configuration</a></span>  
        <span id="nav-examples"><a href="/examples/">Examples</a></span>
```



Apache Tomcat/9.0.89

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Developer Quick Start

Managing Tomcat

Documentation

Getting Help

Other Downloads

Other Documentation

Get Involved

Miscellaneous

Apache Software Foundation

3. Create a Vagrantfile to perform the following:

Requirements:

- Image: bento/ubuntu-24.04
- Guest port 7070 → Host port 9090
- Memory: 2GB
- Java 8
- Tomcat 9

Vagrantfile

NOTE : I used shell script as inline:

```
● ● ●

1 Vagrant.configure("2") do |config|
2   config.vm.box = "bento/ubuntu-24.04"
3
4   config.vm.network "forwarded_port", guest: 7070, host: 9090
5
6   config.vm.provider "virtualbox" do |vb|
7     vb.memory = "2048"
8   end
9
10  config.vm.provision "shell", inline: <<-SHELL
11    apt update
12    apt install -y openjdk-8-jdk wget
13
14    cd /opt
15    wget https://archive.apache.org/dist/tomcat/tomcat-9/v9.0.89/bin/apache-tomcat-9.0.89.tar.gz
16    tar -xvf apache-tomcat-9.0.89.tar.gz
17    mv apache-tomcat-9.0.89 tomcat9
18
19    sed -i 's/port="8080"/port="7070"/' /opt/tomcat9/conf/server.xml
20    /opt/tomcat9/bin/startup.sh
21  SHELL
22 end
23
```

➤ Run Vagrantfile

`vagrant up` & to login vm use: `vagrant ssh`

➤ Access from host:

<http://localhost:9090>

The screenshot displays two windows side-by-side. On the left is a terminal window titled 'vagrant@vagrant:' showing Tomcat startup logs and system information. On the right is a web browser window titled 'Apache Tomcat/9.0.89' showing the Tomcat 9.0.89 homepage.

Terminal Log (Left):

```
default: apache-tomcat-9.0.89/webapps/manager/WEB-INF/jsp/connectorTrust.edCerts.jsp
default: apache-tomcat-9.0.89/webapps/manager/WEB-INF/jsp/sessionDetail.jsp
default: apache-tomcat-9.0.89/webapps/manager/WEB-INF/jsp/sessionsList.jsp
default: apache-tomcat-9.0.89/webapps/manager/WEB-INF/web.xml
default: apache-tomcat-9.0.89/webapps/manager/css/manager.css
default: apache-tomcat-9.0.89/webapps/manager/images/asf-logo.svg
default: apache-tomcat-9.0.89/webapps/manager/images/tomcat.svg
default: apache-tomcat-9.0.89/webapps/manager/index.jsp
default: apache-tomcat-9.0.89/webapps/manager/status.xsd
default: apache-tomcat-9.0.89/webapps/manager/xform.xsl
default: apache-tomcat-9.0.89/bin/catalina.sh
default: apache-tomcat-9.0.89/bin/ciphers.sh
default: apache-tomcat-9.0.89/bin/configtest.sh
default: apache-tomcat-9.0.89/bin/daemon.sh
default: apache-tomcat-9.0.89/bin/digest.sh
default: apache-tomcat-9.0.89/bin/makebase.sh
default: apache-tomcat-9.0.89/bin/setclasspath.sh
default: apache-tomcat-9.0.89/bin/shutdown.sh
default: apache-tomcat-9.0.89/bin/startup.sh
default: apache-tomcat-9.0.89/bin/tool-wrapper.sh
default: apache-tomcat-9.0.89/bin/version.sh
default: Tomcat started.

D:\vagrant-vms>vagrant ssh
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-86-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Wed Feb  4 10:08:09 PM UTC 2026

System load:          0.0
Usage of /:           16.6% of 30.34GB
Memory usage:         18%
Swap usage:          0%
Processes:            144
Users logged in:     0
IPv4 address for eth0: 10.0.2.15
IPv6 address for eth0: fd17:625c:f037:2:a00:27ff:fe8:c2eb
```

Tomcat Home Page (Right):

The Apache Tomcat 9.0.89 homepage features a central message: "If you're seeing this, you've successfully installed Tomcat. Congratulations!" Below this is a cartoon cat logo. To the right, there's a sidebar with "Recommended Reading" links: [Security Considerations How-To](#), [Manager Application How-To](#), and [Clustering/Session Replication How-To](#). The main content area includes sections for "Developer Quick Start" (links to Tomcat Setup, First Web Application, Realms & AAA, JDBC DataSources, Examples, and Servlet Specifications), "Managing Tomcat" (information about manager webapp access and users), "Documentation" (links to Tomcat 9.0 Documentation, Configuration, and Wiki), and "Getting Help" (links to FAQ and Mailing Lists, along with descriptions of various mailing lists like tomcat-announce, tomcat-users, taglibs-user, taglibs-dev, and tomcat-dev).

DevOps Section:

1. Read about Docker: What it is, use cases, and basic commands?

⇒ Docker is a containerization platform that allows applications to be packaged with all their dependencies and run consistently across different environments.

Why Docker is used:

- ❖ Faster deployments
- ❖ Environment consistency
- ❖ Lightweight compared to virtual machines
- ❖ Easy scaling and portability

Common use cases:

- ❖ Microservices architecture
- ❖ CI/CD pipelines
- ❖ Application isolation
- ❖ Dev/Test environments

Basic Docker Commands:

docker –version

docker login

docker pull <image_name>

docker push <image_name>

docker images

docker ps

docker ps -a

docker build -t myapp .

docker run -d -p 80:80 nginx

docker stop <container_id>

docker rm <container_id>

docker rmi <image_id>

docker logs container_name

docker network ls

2. Create a Dockerfile for a Tomcat Docker image, deploy a sample WAR in it, and push it to a Docker registry?

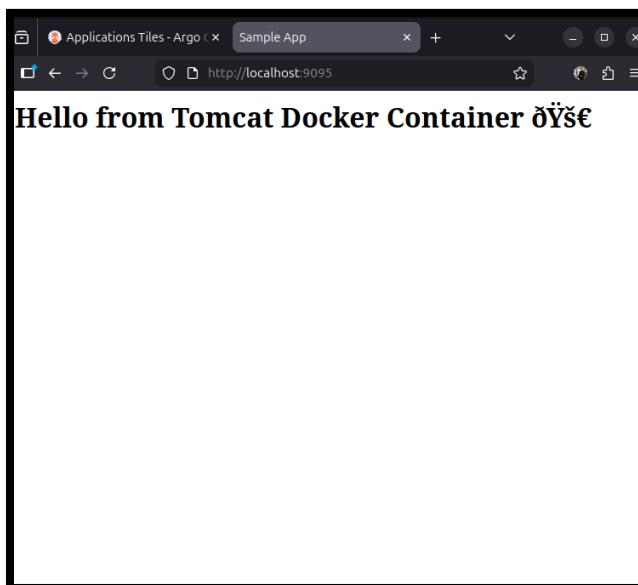
Dockerfile

```
FROM tomcat:9-jdk8

RUN rm -rf /usr/local/tomcat/webapps/*
COPY sample.war /usr/local/tomcat/webapps ROOT.war
EXPOSE 8080
CMD ["catalina.sh", "run"]
```

```
mohamed@master:~/Desktop/tomcat-docker/sample-app$ cd tomcat-docker/
mohamed@master:~/Desktop/tomcat-docker$ ls
Dockerfile sample-app
mohamed@master:~/Desktop/tomcat-docker$ cd sample-app$
mohamed@master:~/Desktop/tomcat-docker/sample-app$ ls
pom.xml src
mohamed@master:~/Desktop/tomcat-docker/sample-app$ mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.example:sample >-----
[INFO] Building sample 1.0
[INFO] -----[ war ]-----
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/2.5/maven-clean-plugin-2.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/2.5/maven-clean-plugin-2.5.pom (3.9 kB at 3.5 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/22/maven-plugins-22.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/22/maven-plugins-22.pom (13 kB at 58 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/21/maven-parent-21.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/21/maven-parent-21.pom (26 kB at 120 kB/s)
```

```
mohamed@master:~/Desktop/tomcat-docker/sample-app$ mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/10.0.1/guava-10.0.1.jar (1.5 MB at 447 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/sisu/sisu-guice/3.1.0/sisu-guice-3.1.0-no_aop.jar (357 kB at 105 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/sisu/org.eclipse.sisu.inject/0.0.0.M2a/org.eclipse.sisu.inject-0.0.0.M2a.jar (202 kB at 59 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/asm/asm/3.3.1/asm-3.3.1.jar (44 kB at 12 kB/s)
[INFO] Packaging webapp
[INFO] Assembling webapp [sample] in [/home/mohamed/Desktop/tomcat-docker/sample-app/target/sample-1.0]
[INFO] Processing war project
[INFO] Copying webapp resources [/home/mohamed/Desktop/tomcat-docker/sample-app/src/main/webapp]
[INFO] Building war: /home/mohamed/Desktop/tomcat-docker/sample-app/target/sample-1.0.war
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 30.766 s
[INFO] Finished at: 2026-02-05T16:52:58+02:00
[INFO] -----
mohamed@master:~/Desktop/tomcat-docker/sample-app$
```



```
mohamed@master:~/Desktop/tomcat-docker$ curl http://localhost:9095
<html>
<head>
<title>Sample App</title>
</head>
<body>
<h1>Hello from Tomcat Docker Container ☕</h1>
</body>
</html>
mohamed@master:~/Desktop/tomcat-docker$
```

Push image to Docker hub registry



docker login

docker tag my-tomcat-app mohamedshehata18/my-tomcat-app

docker push mohamedshehata18/my-tomcat-app

```
mohamed@master: ~/Desktop/tomcat-docker
c604c35b8778175f4de7a192af4ab48eac43033ad88bde4ca785c9af711a7617
mohamed@master:~/Desktop/tomcat-docker$ curl http://localhost:9095
<html>
<head>
    <title>Sample App</title>
</head>
<body>
    <h1>Hello from Tomcat Docker Container 🚀</h1>
</body>
</html>
mohamed@master:~/Desktop/tomcat-docker$ docker tag tomcat-sample-app mohamedshehata18/tomcat-sample-app:1.0
mohamed@master:~/Desktop/tomcat-docker$ docker push mohamedshehata18/tomcat-sample-app:1.0

The push refers to repository [docker.io/mohamedshehata18/tomcat-sample-app]
94376c6c0557: Pushed
59afa6630963: Pushed
81a72c85106a: Pushing  3.146MB/16.98MB
a6269f81c30f: Pushed
bddfaad40945: Pushed
a3629ac5b9f4: Pushing  1.049MB/29.73MB
3fc7e5b7949d: Pushed
2b400b1da126: Pushing  1.049MB/14.01MB
4f4fb700ef54: Mounted from mohamedshehata18/laravel-docker-app
8532bae8ee50: Pushing  1.049MB/54.74MB
S
```

The screenshot shows a browser window with two tabs. The left tab is a Docker Hub repository page for `mohamedshehata18/tomcat-sample-app`. It shows basic repository information, Docker commands (including a `docker push` command), and a table of pushed images with their sizes. The right tab is a terminal window showing the command history for pushing the image to Docker Hub.

Docker Hub Repository Page:

- Repository Details:** mohamedshehata18/tomcat-sample-app
- Last pushed:** 3 minutes ago
- Repository size:** 110.1 MB
- Tags:** 1.0
- Docker Commands:** `docker push mohamedshehata18/tomcat-sample-app:tagname`
- Pushed Images:**

Tag	OS	Type	Pulled	Pushed
1.0	Linux	Image	Less than 1 day	4 minutes

Terminal History (Right Tab):

```
mohamed@master: ~/Desktop/tomcat-docker
<body>
    <h1>Hello from Tomcat Docker Container 🚀</h1>
</body>
</html>
mohamed@master:~/Desktop/tomcat-docker$ docker tag tomcat-sample-app mohamedshehata18/tomcat-sample-app:1.0
mohamed@master:~/Desktop/tomcat-docker$ docker push mohamedshehata18/tomcat-sample-app:1.0

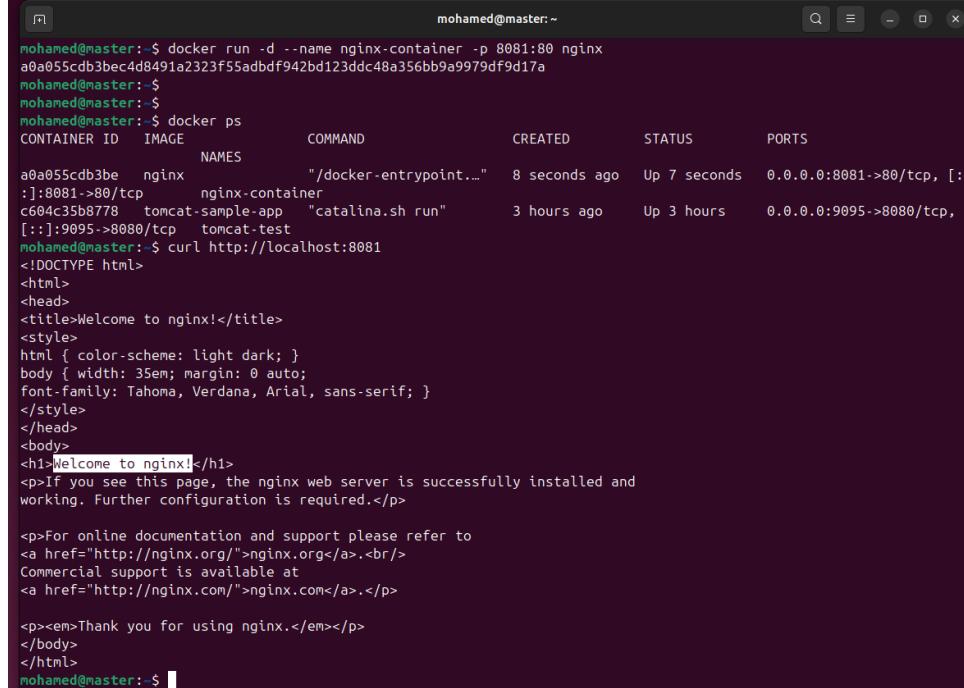
The push refers to repository [docker.io/mohamedshehata18/tomcat-sample-app]
94376c6c0557: Pushed
59afa6630963: Pushed
81a72c85106a: Pushing  16.98MB/16.98MB
a6269f81c30f: Pushed
bddfaad40945: Pushed
81a72c85106a: Pushing  16.98MB/16.98MB
3fc7e5b7949d: Pushed
81a72c85106a: Pushing  16.98MB/16.98MB

81a72c85106a: Pushed

a3629ac5b9f4: Pushing  11.53MB/29.73MB
2b400b1da126: Pushing  12.58MB/14.01MB
2b400b1da126: Pushed
8532bae8ee50: Pushing  17.83MB/54.74MB
8532bae8ee50: Pushed
1.0: digest: sha256:e33c56ac2a8845bf982c759b9ff4d507bf3440dd19e9c1477c0d279566d8c8c8 size: 856
```

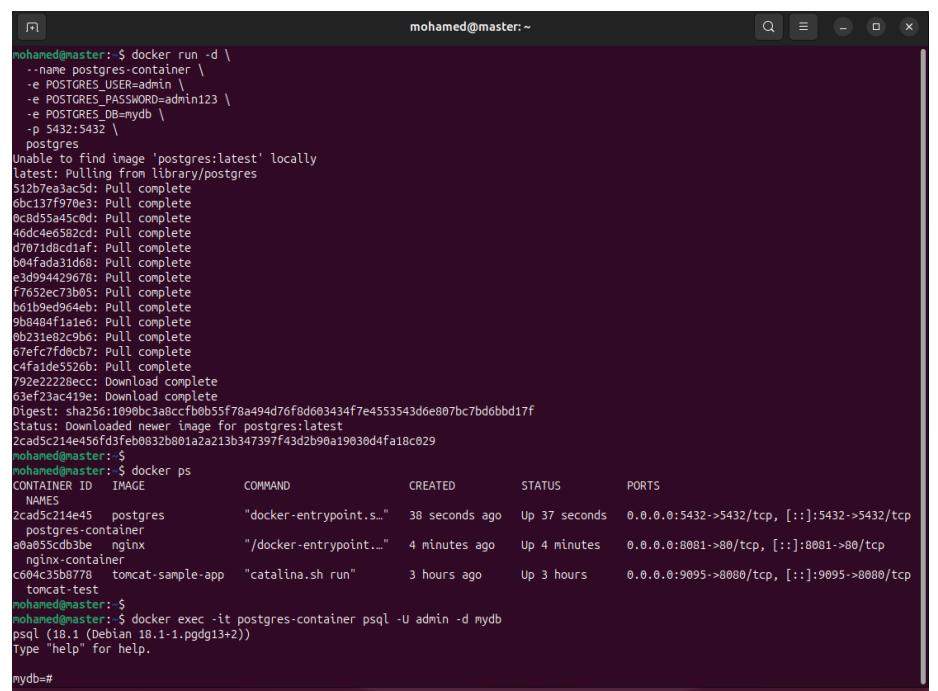
3- Run Nginx & PostgreSQL Containers

```
docker run -d --name nginx-container -p 8081:80 nginx
```



```
mohamed@master:~$ docker run -d --name nginx-container -p 8081:80 nginx
a0a055cdb3be4cd8491a2323f55adbd942bd123ddc48a356bb9a9979df9d17a
mohamed@master:~$ 
mohamed@master:~$ 
mohamed@master:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
 NAMES
a0a055cdb3be nginx "/docker-entrypoint..." 8 seconds ago Up 7 seconds 0.0.0.0:8081->80/tcp, [::]:8081->80/tcp
nginx-container
c604c35b8778 tomcat-sample-app "catalina.sh run" 3 hours ago Up 3 hours 0.0.0.0:9095->8080/tcp, [::]:9095->8080/tcp
tomcat-test
mohamed@master:~$ curl http://localhost:8081
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">http://nginx.org/</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">http://nginx.com/</a>.</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
mohamed@master:~$
```

```
docker run -d \
--name postgres-container \
-e POSTGRES_USER=admin \
-e POSTGRES_PASSWORD=admin123 \
-e POSTGRES_DB=mydb \
-p 5432:5432 \
postgres
```



```
mohamed@master:~$ docker run -d \
--name postgres-container \
-e POSTGRES_USER=admin \
-e POSTGRES_PASSWORD=admin123 \
-e POSTGRES_DB=mydb \
-p 5432:5432 \
postgres
Unable to find image 'postgres:latest' locally
latest: Pulling from library/postgres
512b7ea3ac5d: Pull complete
6bc137f970e3: Pull complete
ec8d55a45c0d: Pull complete
46dc4e6582cd: Pull complete
d771f08cd1af: Pull complete
b04fadad31d6b: Pull complete
e3d994429678: Pull complete
f7652ec73b05: Pull complete
b61b9ed964eb: Pull complete
9b6484f1a1ed: Pull complete
0b231e02c9b6: Pull complete
67fcfc7f0dc0b7: Pull complete
c4fa1de5526b: Pull complete
792e22220ecc: Download complete
63ef23ac419e: Download complete
Digest: sha256:1090b0c3abccfb0b5f78a494d76f0d603434f7e4553543d6e087bc7bd6bbd17f
Status: Downloaded newer image for postgres:latest
2cad5c214e456fd3febe0832b001a2a213b347397f43d2b90a19030d4fa18c029
mohamed@master:~$ 
mohamed@master:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
 NAMES
2cad5c214e45 postgres "docker-entrypoint.s..." 38 seconds ago Up 37 seconds 0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp
postges-container
a0a055cdb3be nginx "/docker-entrypoint..." 4 minutes ago Up 4 minutes 0.0.0.0:8081->80/tcp, [::]:8081->80/tcp
nginx-container
c604c35b8778 tomcat-sample-app "catalina.sh run" 3 hours ago Up 3 hours 0.0.0.0:9095->8080/tcp, [::]:9095->8080/tcp
tomcat-test
mohamed@master:~$ 
mohamed@master:~$ docker exec -it postgres-container psql -U admin -d mydb
psql (18.1 (Debian 18.1.1-ppgdg13+2))
Type "help" for help.

mydb=#
```

Kubernetes Section:

Kubernetes (K8s) is a container orchestration platform used to manage, scale, and deploy containerized applications automatically.

It handles:

- Pod scheduling
 - Service discovery
 - Scaling
 - Self-healing
 - Load balancing
-

2. Explain the purpose of master and worker nodes, and how to identify whether a node is master or worker.

Master Node (Control Plane)

The master node is the brain of the Kubernetes cluster. It manages, controls, and makes decisions about the cluster.

Worker Node

Worker nodes run containerized applications and are managed by the master node.

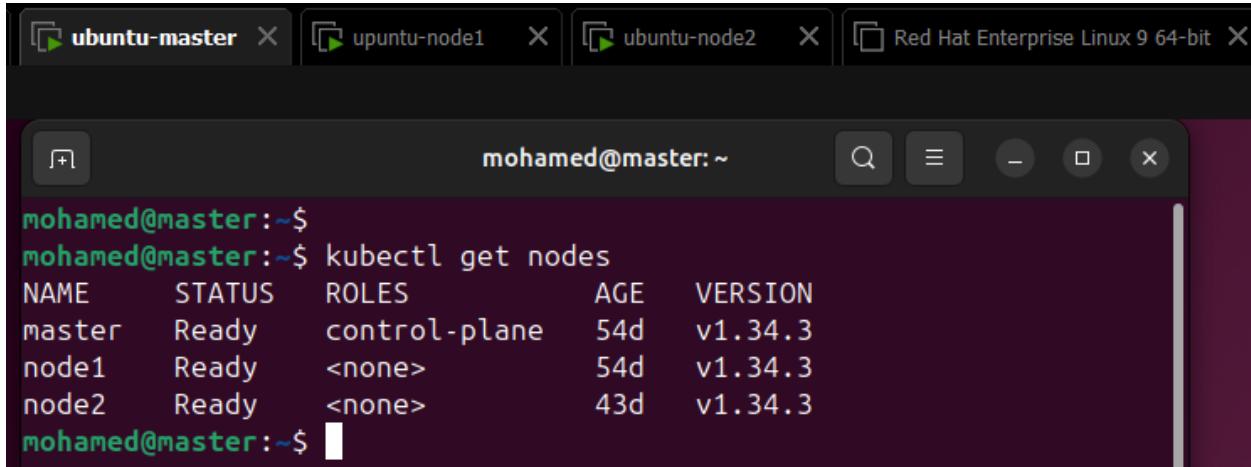
Key Components (run on master)

- kube-apiserver → Entry point for all cluster operations
- etcd → Stores cluster state & configuration
- kube-scheduler → Decides where pods run
- kube-controller-manager → Ensures desired state

Key Components (run on Worker)

- Kubelet: Agent that runs on each node, ensures containers are running in pods
- Kube-proxy: Network proxy that maintains network rules for pod communication
- Container Runtime: Software responsible for running containers

Q: How to Identify Master vs Worker Nodes ?

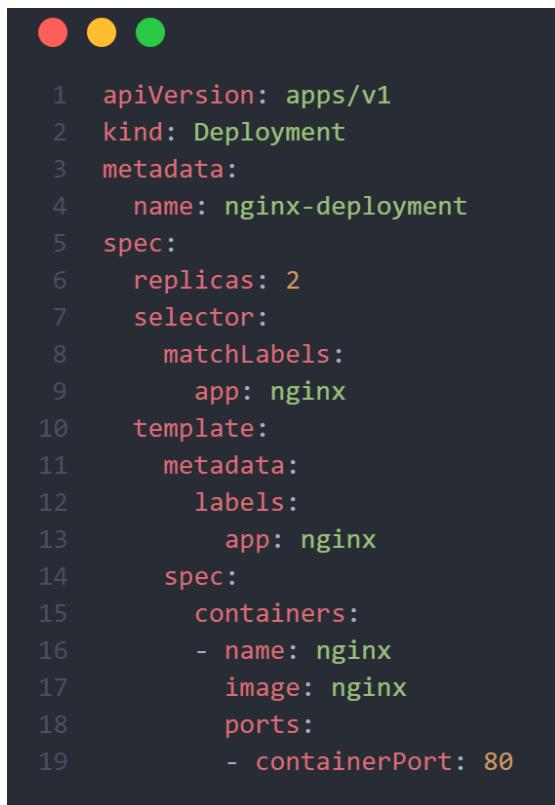


```
mohamed@master:~$ kubectl get nodes
NAME     STATUS    ROLES      AGE     VERSION
master   Ready     control-plane   54d    v1.34.3
node1    Ready     <none>    54d    v1.34.3
node2    Ready     <none>    43d    v1.34.3
mohamed@master:~$
```

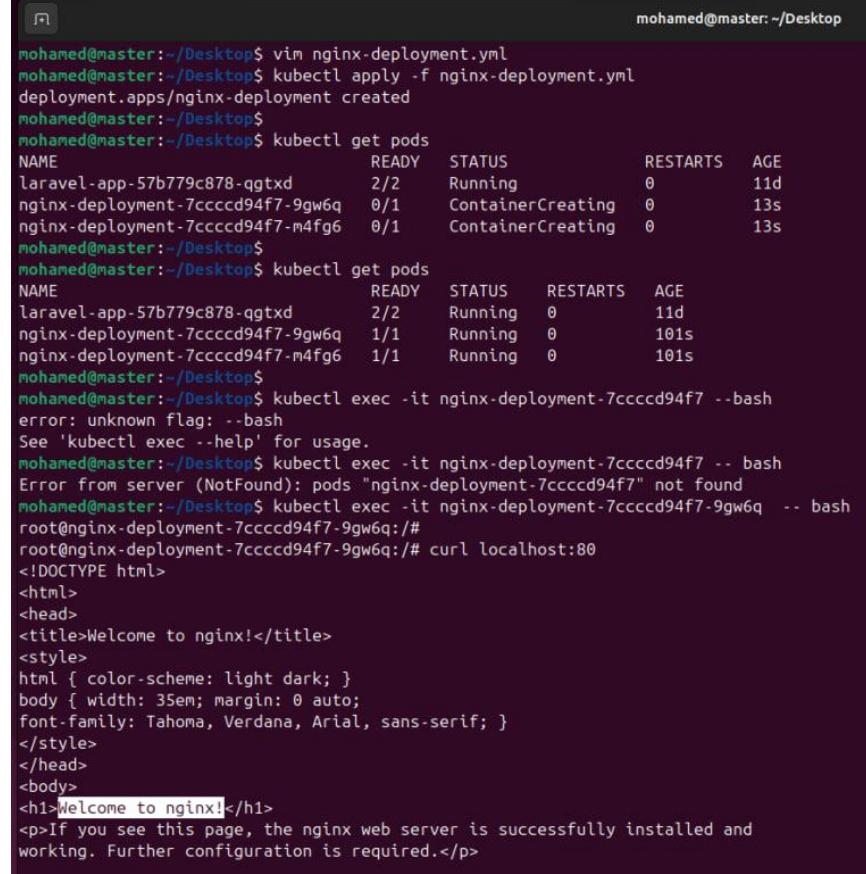
3. Try installing K8s, K3s, or MicroK8s and perform a sample deployment.

I install K8S using Kubeadm on vmware used 1 Master node, 2 Worker nodes

Sample Kubernetes Deployment:



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx
          ports:
            - containerPort: 80
```



```
mohamed@master:~/Desktop$ vim nginx-deployment.yml
mohamed@master:~/Desktop$ kubectl apply -f nginx-deployment.yml
deployment.apps/nginx-deployment created
mohamed@master:~/Desktop$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
laravel-app-57b779c878-qgtxd   2/2     Running   0          11d
nginx-deployment-7cccd94f7-9gw6q   0/1     ContainerCreating   0          13s
nginx-deployment-7cccd94f7-m4fg6   0/1     ContainerCreating   0          13s
mohamed@master:~/Desktop$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
laravel-app-57b779c878-qgtxd   2/2     Running   0          11d
nginx-deployment-7cccd94f7-9gw6q   1/1     Running   0          101s
nginx-deployment-7cccd94f7-m4fg6   1/1     Running   0          101s
mohamed@master:~/Desktop$ kubectl exec -it nginx-deployment-7cccd94f7 -- bash
error: unknown flag: --bash
See 'kubectl exec --help' for usage.
mohamed@master:~/Desktop$ kubectl exec -it nginx-deployment-7cccd94f7 -- bash
Error from server (NotFound): pods "nginx-deployment-7cccd94f7" not found
mohamed@master:~/Desktop$ kubectl exec -it nginx-deployment-7cccd94f7-9gw6q -- bash
root@nginx-deployment-7cccd94f7-9gw6q:/# curl localhost:80
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto; font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
```

Key Concepts:

1. RAID (Redundant Array of Independent Disks)

RAID is a technology used to combine multiple physical hard disks into one logical unit to improve performance, fault tolerance, or both.

Why RAID is used

- Protect data from disk failure
- Increase read/write performance
- Improve system availability

Use cases

- Databases (Oracle, PostgreSQL)
- File servers
- Critical production systems

Common RAID Levels

- **RAID 0 (Striping)**
 - Splits data across disks
 - High performance
 - No fault tolerance (if one disk fails → data lost)
- **RAID 5 (Striping + Parity)**
 - Data + parity distributed across disks
 - Can tolerate **one disk failure**
 - Good balance between performance and safety
- **RAID 1 (Mirroring)**
 - Data is duplicated on two disks
 - High data protection
 - Uses more disk space
- **RAID 10 (1+0)**
 - Combination of RAID 1 and RAID 0
 - High performance + high availability
 - Used in databases and production systems

2. DevOps

DevOps is a culture and set of practices that aims to **bridge the gap between Development and Operations** to deliver software faster, more reliably, and with better quality.

Key goals of DevOps

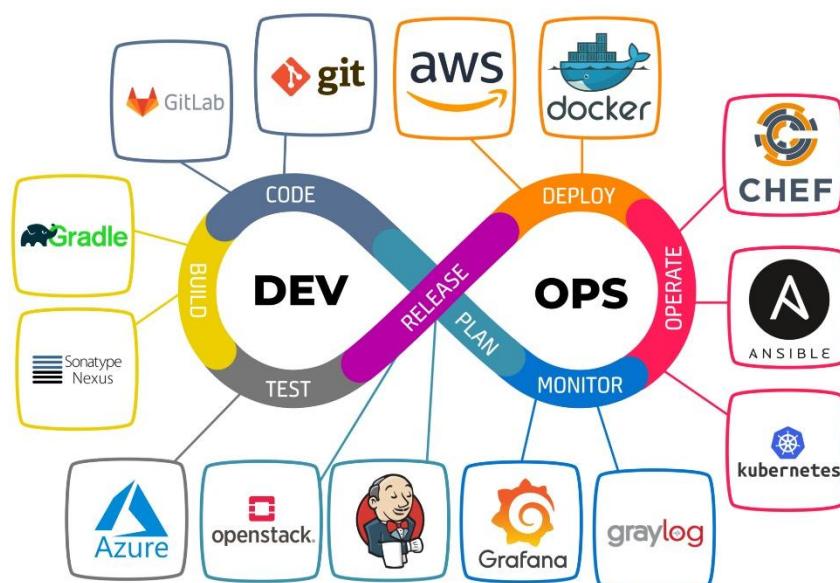
- Faster deployments
- Automation of repetitive tasks
- Continuous delivery
- High system reliability

Benefits

- Reduced human errors
- Faster release cycles
- Better collaboration between teams
- Improved system stability

Core DevOps practices

- **CI/CD** (Continuous Integration & Continuous Deployment)
- **Infrastructure as Code (IaC)** – Terraform, Ansible
- **Containerization** – Docker
- **Orchestration** – Kubernetes
- **Monitoring & Logging** – Prometheus, Grafana, ELK



3. High Availability (HA) and Disaster Recovery (DR) – what they are, and technologies that can be used for them (examples: Oracle, PostgreSQL, or Tomcat)

- High Availability (HA)

HA ensures that a system remains operational with minimal downtime, even if a component fails.

HA techniques	Examples
<ul style="list-style-type: none">• Load balancing• Active/Active or Active/Passive setups• Failover clustering• Replication	<ul style="list-style-type: none">• PostgreSQL: Streaming replication, Patroni• Oracle: Oracle RAC• Tomcat: Multiple Tomcat instances behind Nginx/HAProxy

Disaster Recovery (DR)

DR focuses on recovering systems and data after a major failure, such as:

- Data center outage
- Natural disasters
- Cyberattacks

DR strategies

- Regular backups
- Offsite backups
- Database snapshots
- Standby environments

Key metrics

- RTO (Recovery Time Objective) – how fast recovery happens
- RPO (Recovery Point Objective) – how much data loss is acceptable

4. Cloud Computing

Cloud Computing is the delivery of computing services (servers, storage, databases, networking, software) over the internet.

Cloud deployment models

- **Public Cloud** – AWS, Azure, GCP
 - **Private Cloud** – On-prem or private data center
 - **Hybrid Cloud** – Mix of public + private
-

Cloud Service Models

IaaS (Infrastructure as a Service)	PaaS (Platform as a Service)	SaaS (Software as a Service)
<ul style="list-style-type: none">• You manage: OS, runtime, apps• Provider manages: hardware, network• Examples:<ul style="list-style-type: none">◦ AWS EC2◦ Azure Virtual Machines	<ul style="list-style-type: none">• You manage: application & data• Provider manages: OS, runtime, infrastructure• Examples:<ul style="list-style-type: none">◦ AWS Elastic Beanstalk	<ul style="list-style-type: none">• Fully managed software• No infrastructure management• Examples:<ul style="list-style-type: none">◦ Gmail◦ Google Drive◦ Salesforce

5. DNS and Load Balancers

DNS (Domain Name System)	Example
DNS translates human-readable domain names into IP addresses.	www.example.com → 192.168.1.1

Load Balancer

A Load Balancer distributes incoming traffic across multiple servers to:

- **Improve performance**
- **Prevent server overload**
- **Increase availability**

Types of Load Balancers

- Layer 4 (Transport level) – TCP/UDP
 - Example: HAProxy (TCP mode)
- Layer 7 (Application level) – HTTP/HTTPS
 - Example: Nginx, AWS ALB

Common tools

- Nginx
- HAProxy
- AWS ELB / ALB
- Kubernetes Service & Ingress