**1.jmeter/**

**2.selenium/**

**3.wireshark/**

**4.dcl/**

**5.openmp/**

**6.mpi library/**

**7.2d**

**8.3d**

**9.weka R**

**JMETER**

**Aim:**

To evaluate the performance of a given web application by conducting load and stress testing using Apache JMeter, a widely-used performance testing tool.

**Tool Description:**

* **Apache JMeter** is an open-source tool designed to test the performance of both static and dynamic resources (such as web applications, databases, and services). It is used to simulate a heavy load on a server, group of servers, network, or object to test its strength or analyze overall performance under different load types.

**Procedure:**

1. **Install Apache JMeter:**
   * Download and install JMeter from [JMeter's official website](https://jmeter.apache.org/).
2. **Launch JMeter:**
   * Navigate to the JMeter installation directory and run the jmeter.bat (Windows) or jmeter (Linux/Mac) file to start JMeter.
3. **Create a Test Plan:**
   * Open JMeter, and in the **Test Plan** window:
     + Right-click on **Test Plan** → Add → **Threads (Users)** → **Thread Group**.
     + Configure the number of threads (users), ramp-up period, and loop count to simulate concurrent users.
4. **Add HTTP Request Sampler:**
   * Right-click on **Thread Group** → Add → **Sampler** → **HTTP Request**.
   * Set the **Server Name or IP** field to the domain of the web application you are testing.
   * Set the **Path** to the specific resource or endpoint (e.g., /home, /login).
5. **Add HTTP Request Defaults (Optional):**
   * To avoid repeating details for each request, add HTTP request defaults by right-clicking on **Test Plan** → Add → **Config Element** → **HTTP Request Defaults**. Set the common server name and protocol here.
6. **Add Listeners:**
   * To view results, add listeners:
     + Right-click on **Thread Group** → Add → **Listener** → choose **View Results in Table**, **View Results in Tree**, or **Aggregate Report** for graphical and statistical output.
7. **Configure and Run the Test:**
   * Set the load parameters (number of users, ramp-up period) based on the expected traffic or stress you want to simulate.
   * Save the Test Plan.
   * Click the **Start** button (green triangle) to run the test.
8. **Analyze Results:**
   * After execution, analyze the results using the listeners. Metrics like average response time, throughput, error rate, and time to first byte are displayed.
   * The **Aggregate Report** will summarize key metrics such as:
     + **Label:** Request name.
     + **# Samples:** Number of requests.
     + **Average Response Time:** Average time taken to process a request.
     + **Throughput:** Requests per second handled by the server.
     + **Error %:** Percentage of failed requests.

**Sample Input/Output:**

* **Input:**
  + Web Application URL: http://example.com
  + Number of users: 100
  + Ramp-up period: 30 seconds
  + Loop count: 10
  + HTTP Request Method: GET
* **Expected Output:**
  + **Average Response Time:** 150 ms
  + **Error Rate:** 1%
  + **Throughput:** 65 requests/second

**Result:**

The performance of the given web application was tested using JMeter. The results indicate that the average response time is within acceptable limits under moderate load, with a low error rate and consistent throughput. These metrics help identify potential bottlenecks and guide optimizations to improve the web application's performance.

**SELENIUM**

**Aim:**

To perform acceptance testing for the web application [**https://the-internet.herokuapp.com/login**](https://the-internet.herokuapp.com/login) using Selenium WebDriver and validate the login functionality using different test cases.

**Tool Description:**

* Selenium allows developers and testers to automate the testing of web pages across different browsers. Selenium web driver allows various prog lang for test script development. Open source

**Procedure:**

1. **Set up the Environment:**
   * Install **Selenium** and **Webdriver Manager** using the following commands:

pip install selenium

pip install webdriver-manager

* + Install **ChromeDriver** via Webdriver Manager for interacting with the Chrome browser.

1. **Test Case Design:**
   * Create a set of login test cases, with different combinations of username and password:
     + Valid login: Correct username and password.
     + Invalid login: Various combinations of incorrect usernames and/or passwords.
     + Empty fields: No input in the username and/or password fields.
2. **Write the Selenium Script:**
   * Use Selenium WebDriver to automate the following actions:
     + Open the login page at [**https://the-internet.herokuapp.com/login**](https://the-internet.herokuapp.com/login).
     + Fill in the login form with the specified username and password for each test case.
     + Click the login button.
     + Verify if the login is successful or failed by checking the presence of the success message.
3. **Run the Selenium Script:**
   * Execute the script and observe the results for each test case.
   * Output the result (success or failure) for each test based on whether the correct message appears.

**Sample Input/Output:**

**Input:**

The script tests the following login credentials:

1. **Username**: "Dinesh", **Password**: "SuperSecretPassword!"
2. **Username**: "Dinesh", **Password**: "WrongPassword"
3. **Username**: "WrongUser", **Password**: "SuperSecretPassword!"
4. **Username**: "tomsmith", **Password**: "SuperSecretPassword!" (valid)

**Output:**

For each test case, the script prints the result of whether the login attempt was successful or failed.

**WIRESHARK TSHARK**

**Aim:**

To analyze network traffic using **Wireshark** and **Nmap** by capturing and inspecting packet data on a local network.

**Tool Description:**

1. **Wireshark**: Wireshark is a popular open-source tool for network protocol analysis. It captures network packets in real-time and displays them in a human-readable format, allowing for in-depth analysis of network communication.
2. **Nmap**: Nmap (Network Mapper) is an open-source tool for network discovery and security auditing. It can perform host discovery, port scanning, and packet manipulation to analyze the network traffic.
3. **TShark**: TShark is the command-line version of Wireshark, capable of capturing and analyzing network traffic from the terminal.

**Procedure:**

1. **Install Wireshark and Nmap**:
   * Download and install **Wireshark** from https://www.wireshark.org/download.html.
   * Download and install **Nmap** from https://nmap.org/download.html.
2. **Get the Local Network IP Address**:
   * Open the command prompt (cmd) and run the following command to get your machine’s local IPv4 address:

bash

Copy code

ipconfig

* + Note down the **IPv4 address** (e.g., 192.168.1.36).

1. **Initiate a Network Scan using Nmap**:
   * Open **Nmap** from the command prompt (cmd) and run the following command to scan TCP traffic on port 80 for your IPv4 address:

bash

Copy code

nping --tcp -p 80 192.168.1.36

* + This will simulate network traffic on TCP port 80.

1. **Open Wireshark to Capture Traffic**:
   * In Wireshark, capture the live traffic by selecting the appropriate network interface (e.g., Wi-Fi or Ethernet).
   * Alternatively, you can use **TShark** (Wireshark's command-line version) to capture packets.
2. **Capture Traffic using TShark**:
   * Open the command prompt and list available network interfaces using:

bash

Copy code

tshark -D

* + Identify the correct interface number (e.g., Wi-Fi) and start capturing packets using:

bash

Copy code

tshark -i 5 -w capture.pcap

This will capture traffic on the specified interface and save it to a file named capture.pcap.

1. **Stop the Capture**:
   * After a few moments of capturing, stop the **TShark** capture by pressing **Ctrl+C** in the command prompt.
   * The captured packets are saved in the capture.pcap file.
2. **Inspect the Captured Traffic**:
   * Open **Wireshark** and load the capture.pcap file.
   * In Wireshark, filter the traffic by specifying the IP address in the filter bar:

bash

Copy code

ip.addr==192.168.1.36

* + Explore the highlighted packets to analyze the communication, including TCP handshakes, HTTP requests, and responses.

1. **Perform a Different Scan with Nmap**:
   * You can run a different Nmap scan on another port, for example, port 90:

bash

Copy code

nping --tcp -p 90 192.168.1.36

* + Capture the network traffic for this scenario using TShark again:

bash

Copy code

tshark -i 5 -w capture.pcap

1. **View Results in Wireshark**:
   * Load the new capture.pcap file in Wireshark.
   * Apply relevant filters to analyze the captured traffic and inspect network communication.

**Sample Input/Output:**

**Input:**

1. **Nmap command**:

bash

Copy code

nping --tcp -p 80 192.168.1.36

1. **Wireshark Filter**:

bash

Copy code

ip.addr==192.168.1.36

**Output:**

* Wireshark captures traffic involving the target IP address.
* Network communication can be seen in detail, including TCP packets, handshakes, and application-layer data.

**Result:**

The network traffic was successfully captured and analyzed using **Wireshark** and **TShark**. The packet captures provided insight into how data flows through the network during a port scan. Various protocols such as TCP were observed, and the traffic involving specific IP addresses was filtered and analyzed.

**OPENMP**

**Aim:**

To compare the performance of sequential and parallel matrix multiplication using OpenMP in a C program.

**Tool Description:**

**OpenMP (Open Multi-Processing):**

* OpenMP is an API that supports multi-platform shared-memory multiprocessing programming in C, C++, and Fortran.
* It enables parallelism by providing directives to the compiler for parallel execution of code.

**Packages to Install:** For Windows using MinGW, you need to install the following packages to enable pthread and math libraries:

1. mingw32-libpthread-old-dll
2. mingw32-libpthreadgc-dev
3. mingw32-libpthreadgc-dll
4. mingw32-libpthreadgce-dev
5. mingw32-libpthreadgce-dll
6. mingw32-libquadmath-dll

**Procedure:**

1. **Install Required Packages:** Install the listed MinGW packages to enable threading and math functionalities.
2. **Write the Source Code:**
   * Include the necessary libraries like stdio.h, stdlib.h, and omp.h.
   * Define matrix multiplication functions for both sequential and parallel methods.
   * Initialize matrices A, B, and C.
   * Time the sequential and parallel execution of matrix multiplication using omp\_get\_wtime().
3. **Compilation Command:** Use the following command to compile the code with OpenMP support:

bash

Copy code

gcc -fopenmp -o output matmul.c

1. **Execution:** Run the executable to compare the performance:

bash

Copy code

output.exe

**Sample Input/Output:**

**Sample Output:**

Copy code

Time taken for sequential matrix multiplication: 12.543218 seconds

Time taken for parallel matrix multiplication: 3.124570 seconds

Element C[0][0]: 2148

Element C[100][100]: 1876

Element C[N-1][N-1]: 1992

**Results:**

The program compares the execution time of matrix multiplication in a sequential manner and a parallelized manner using OpenMP. Typically, the parallel version executes faster, particularly for large matrices, due to the concurrent execution of loops across multiple threads.

**MPI LIBRARY**

**Aim:**

To implement the Floyd-Warshall algorithm using the MPI (Message Passing Interface) library in Python to compute the shortest path matrix of a graph distributed across multiple processes.

**Tool Description:**

**MPI (Message Passing Interface):**

* MPI is a standard for parallel computing that enables multiple processes to communicate with one another by passing messages.
* It is particularly useful for distributed memory architectures, allowing processes to run on different processors or machines.

**MPI Library - mpi4py:**

* mpi4py is a Python wrapper for the MPI library, enabling MPI functionalities in Python.
* Functions like Bcast(), Scatter(), and Gather() are used to distribute and collect data among processes.

**To verify MPI installation:** Use the command mpiexec to check if MPI is installed and running correctly.

**Procedure:**

1. **Install MPI Library:** Install the mpi4py package using the following command:

bash

Copy code

pip install mpi4py

1. **Write the Source Code:**
   * The code implements the Floyd-Warshall algorithm in parallel using MPI.
   * It initializes the MPI environment using MPI.COMM\_WORLD and distributes the matrix rows across multiple processes using Scatter().
   * Each process computes a portion of the shortest path matrix.
   * The results are gathered using Gather(), and the shortest path matrix is printed by the root process.
2. **Compilation/Execution:** Run the code using mpiexec to launch multiple processes:

bash

Copy code

mpiexec -n 4 python file.py

1. **Matrix Input:**
   * Input an adjacency matrix for the graph. Use infinity (inf) to represent no direct path between vertices.
   * For example:

Copy code

0 3 inf 7

8 0 2 inf

5 inf 0 1

2 inf inf 0

**Sample Input/Output:**

**Sample Input:**

Copy code

Enter the number of vertices: 4

Enter the adjacency matrix (use inf for INF):

0 3 inf 7

8 0 2 inf

5 inf 0 1

2 inf inf 0

**Sample Output:**

lua

Copy code

Shortest path matrix:

0 3 5 7

3 0 2 6

3 6 0 1

2 5 7 0

Elapsed time: 0.00012345 seconds

**Synfig**

* **Overview**: Synfig is an open-source 2D vector-based animation software. It is designed to produce high-quality animations using fewer resources and provides an alternative to traditional frame-by-frame animation.
* **Key Features**:
  + **Vector-Based Animation**: Synfig supports vector graphics, allowing smooth animations that scale without loss of quality.
  + **Bones System**: This feature enables complex character animation through skeletons, making it easier to create and manipulate animations with minimal keyframes.
  + **Advanced Controls**: Synfig provides control over every aspect of your animation, including gradients, filters, and distortions, allowing for detailed effects.
  + **Timeline and Keyframes**: It supports a timeline-based workflow, where keyframes can be added and manipulated to create animations.
  + **Cutout Animation**: In addition to vector-based animations, Synfig supports cutout animation techniques, allowing users to animate images or pre-drawn artwork.
* **Use Cases**: Synfig is ideal for creating 2D animations for films, web animations, commercials, educational videos, and more. It’s also commonly used by hobbyists and small animation studios due to its ease of use and zero cost.

**Blender**

* **Overview**: Blender is a powerful open-source 3D creation suite used for modeling, sculpting, rigging, animation, simulation, rendering, compositing, and motion tracking. It is widely used in various industries, including animation, game development, and visual effects.
* **Key Features**:
  + **3D Modeling and Sculpting**: Blender provides robust tools for 3D modeling and sculpting, allowing users to create complex 3D objects, characters, and environments.
  + **Rigging and Animation**: Blender has a powerful rigging system for creating character rigs and supports complex animations, including both skeletal animation and keyframe-based techniques.
  + **Physics Simulation**: It includes tools for simulating realistic physical phenomena such as smoke, fluid, hair, cloth, and rigid body dynamics.
  + **Rendering Engines**: Blender has built-in rendering engines like Cycles and Eevee, enabling both physically-based rendering and real-time previews.
  + **Compositing and Post-Production**: Blender includes a node-based compositing system, making it possible to perform advanced post-production tasks such as color correction and visual effects.
  + **Game Engine**: Although Blender’s native game engine has been deprecated, it can be integrated with other game development workflows, including exporting models for use in game engines like Unity or Unreal Engine.
* **Use Cases**: Blender is used in professional settings for 3D animations, visual effects, video game assets, and even architectural visualization. It’s popular for creating 3D animated films, indie games, and artistic 3D designs. Blender is also widely used in the VR and AR industries due to its extensive toolset and active development community.

3D

Download

Cube will be there

S on x axis big rectangle

Add>mesh>plane

S on plane

G Y grab on y axis

G Z grab on z axis

Add>mesh>cube

S G

Select cube physics>rigid body>20kg>shape>mesh>friction>1>damping>0.3>rotation>0.6

Select plane physic>rigid body>type>passive>shape>mesh>friction>1

Select cuboid physics>rigid body>type>passive>shape>mesh>animated click>friction>0.5

Play

At 120>click cuboid>click N to see location>click I to keyframe>go to 130 time>G Y bring cuboid to front>left click to confirm

File>save as

Click cube>material (right side)>surface>diffuse>shift D to duplicate

Click scene(right side) >rigid body world>cache>bake

**WEKA R**

**Weka**

* **Overview**: Weka is an open-source software suite for machine learning and data mining tasks. It provides a collection of algorithms for data preprocessing, classification, regression, clustering, and association rule mining.
* **Key Features**:
  + **User-Friendly Interface**: Weka has a graphical user interface (GUI) that makes it accessible for users who may not have a programming background.
  + **Wide Range of Algorithms**: It includes numerous machine learning algorithms, including decision trees, neural networks, support vector machines, and more.
  + **Data Preprocessing**: Weka provides tools for data cleaning, normalization, transformation, and visualization.
  + **Model Evaluation**: It allows users to evaluate models using techniques like cross-validation and provides various metrics for performance assessment.
  + **Integration**: Weka can be integrated with other programming languages like Java and can be used in conjunction with other tools.
* **Use Cases**: Weka is commonly used in academic research, teaching, and for prototyping machine learning models in various domains, including healthcare, finance, and marketing.

**R**

* **Overview**: R is a programming language and software environment primarily used for statistical computing and graphics. It is widely used among statisticians, data analysts, and data scientists for data analysis, visualization, and machine learning.
* **Key Features**:
  + **Statistical Analysis**: R offers a wide array of statistical techniques, including linear and nonlinear modeling, time-series analysis, and clustering.
  + **Data Visualization**: R has powerful packages like ggplot2 for creating high-quality graphs and visualizations.
  + **Extensibility**: R has a rich ecosystem of packages (CRAN) that extends its capabilities, allowing users to implement various algorithms and methods.
  + **Community Support**: R has a large and active community, making it easy to find resources, tutorials, and help.
  + **Integration**: R can interface with other languages (like Python and C++), databases, and big data technologies, making it versatile for data manipulation and analysis.
* **Use Cases**: R is used in academia and industry for data analysis, statistical modeling, bioinformatics, machine learning, and data visualization in fields like finance, social sciences, and healthcare.