RMI IMPLEMENTATION REPORT-2023115097(ASIFALEKHA M)

---

1. Introduction

Remote Method Invocation (RMI) is a Java-based distributed computing mechanism that allows a program to invoke methods on an object located in another machine over a network. RMI provides a high-level abstraction for remote communication by allowing method calls to appear as local method calls.

In this project, Java RMI is used to implement a client-server application where the client remotely invokes arithmetic methods hosted on a cloud server.

---

2. Objectives

The main objectives of this RMI implementation are:

- To understand remote object communication.
- To design a remote interface using Java.
- To implement a remote object and register it with the RMI registry.
- To allow clients to invoke methods remotely.
- To host the server application in a cloud environment.

---

3. System Architecture

The system follows a **distributed client-server architecture**.

Components:

- RMI Client (local machine)
- RMI Registry (cloud server)
- Remote Object (cloud server)

Flow:
Client → RMI Registry → Remote Object → Result returned to Client

---

4. Tools and Technologies Used

Component	**Technology**
Programming Language	Java
Cloud Platform	AWS EC2
Operating System	Ubuntu Linux

Component	Technology
JDK	OpenJDK 17
Network Protocol	TCP/IP
Development Environment	VS Code / Terminal

---

5. RMI Server Implementation

5.1 Description

The RMI server defines a remote interface and implements it in a concrete class. The server creates an instance of the remote object and registers it with the RMI registry so that clients can locate it.

5.2 Server Responsibilities

- Define remote methods.
- Create and export remote object.
- Register object with RMI registry.
- Handle client requests.
- Send results back to client.

5.3 Server Features

- Supports multiple clients.
- Uses Java object serialization.
- Provides transparent remote access.
- Runs continuously on cloud server.

---

6. RMI Client Implementation

6.1 Description

The RMI client connects to the remote registry using the server's public IP address and looks up the registered remote object. It then invokes methods on the object.

6.2 Client Responsibilities

- Locate RMI registry.
- Lookup remote object.

- Invoke remote methods.
- Display results.

6.3 Client Features

- Simple command-line interface.
- Supports remote invocation.
- Handles network failures.
- Works from any location.

---

7. Error Handling

The following error handling techniques were used:

- Try-catch blocks to handle RemoteException.
- Handles UnknownHostException.
- Handles connection timeout.
- Prints error messages for debugging.

---

8. Cloud Hosting Details

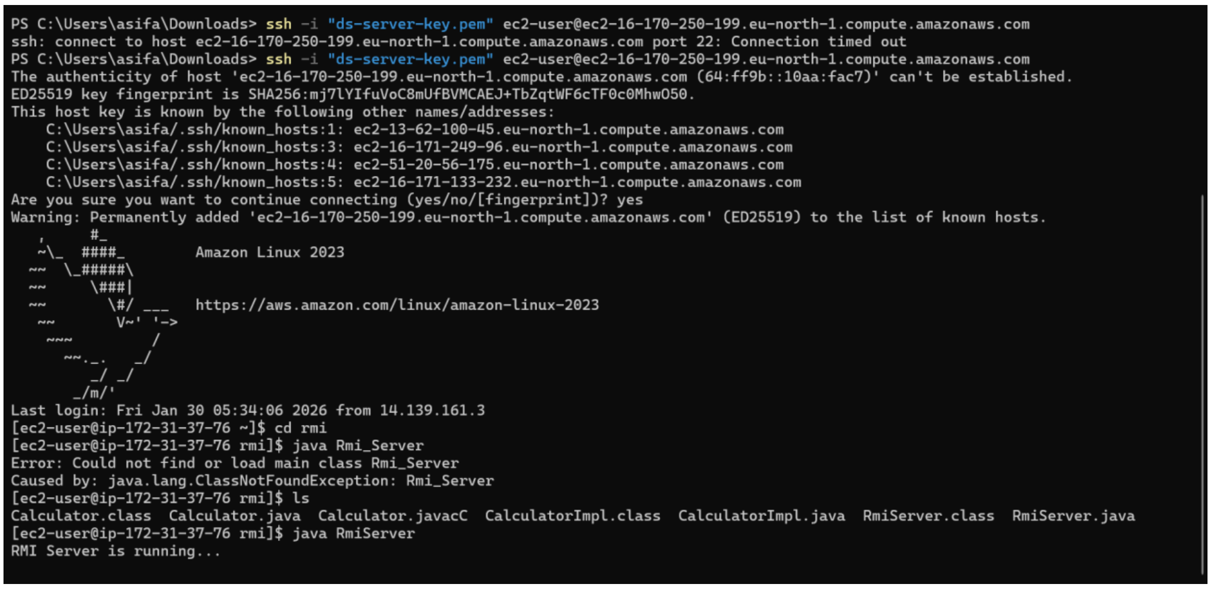
The RMI server was deployed on an AWS EC2 instance.

Parameter	**Value**
Instance Type	t2.micro
OS	Ubuntu 22.04
Open Ports	22 (SSH), 1099 (RMI)
Access Method	Public IPv4

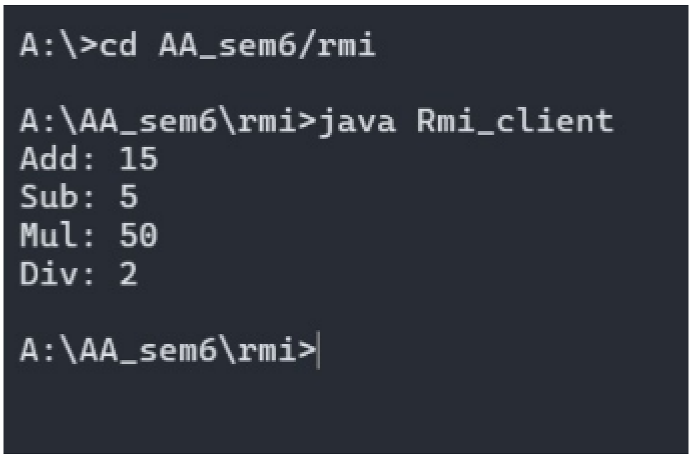
The client runs on the local system and connects to the server using the public IP address.

---

9. Expected Output



```
PS C:\Users\asifa\Downloads> ssh -i "ds-server-key.pem" ec2-user@ec2-16-170-250-199.eu-north-1.compute.amazonaws.com
ssh: connect to host ec2-16-170-250-199.eu-north-1.compute.amazonaws.com port 22: Connection timed out
PS C:\Users\asifa\Downloads> ssh -i "ds-server-key.pem" ec2-user@ec2-16-170-250-199.eu-north-1.compute.amazonaws.com
The authenticity of host 'ec2-16-170-250-199.eu-north-1.compute.amazonaws.com (64:ff9b::10aa:fac7)' can't be established.
ED25519 key fingerprint is SHA256:mj7lYIfuVoC8mUfBVMCAEJ+TbZqtWF6cTF0c0Mhw050.
This host key is known by the following other names/addresses:
    C:\Users\asifa/.ssh/known_hosts:1: ec2-13-62-100-45.eu-north-1.compute.amazonaws.com
    C:\Users\asifa/.ssh/known_hosts:3: ec2-16-171-249-96.eu-north-1.compute.amazonaws.com
    C:\Users\asifa/.ssh/known_hosts:4: ec2-51-20-56-175.eu-north-1.compute.amazonaws.com
    C:\Users\asifa/.ssh/known_hosts:5: ec2-16-171-133-232.eu-north-1.compute.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-16-170-250-199.eu-north-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Fri Jan 30 05:34:06 2026 from 14.139.161.3
[ec2-user@ip-172-31-37-76 ~]$ cd rmi
[ec2-user@ip-172-31-37-76 rmi]$ java Rmi_Server
Error: Could not find or load main class Rmi_Server
Caused by: java.lang.ClassNotFoundException: Rmi_Server
[ec2-user@ip-172-31-37-76 rmi]$ ls
Calculator.class  Calculator.java  Calculator.javacC  CalculatorImpl.class  CalculatorImpl.java  RmiServer.class  RmiServer.java
[ec2-user@ip-172-31-37-76 rmi]$ java RmiServer
RMI Server is running...
```



```
A:\>cd AA_sem6/rmi

A:\AA_sem6\rmi>java Rmi_client
Add: 15
Sub: 5
Mul: 50
Div: 2

A:\AA_sem6\rmi>
```

Conclusion

This RMI implementation demonstrates how distributed objects can communicate over a network. The project successfully shows remote method invocation using Java and cloud deployment, fulfilling all assignment requirements.