Team Name: **TECHNOZ**

Problem Statement: **METHOD TRACE ANALYZER**

Method trace has the functionality to –

* Trace entry to and exit from Java methods run by the JVM.
* Trace stacktrace for methods.
* Filter the methods to be traced based on classname, method name, or both. It also supports wildcards to create complex method selections. -

Role of Members:

RAHUL KUMAR: CODING

VAIBHAV PALIWAL: GUI MAKING

SHRUTI AGARWAL: SEARCHING

VISHAL TULSANI: DOCUMENTATION

Scope of Work and Technology Used:

**Java:**

**Java** is a general-purpose [computer-programming language](https://en.wikipedia.org/wiki/Programming_language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming),[[15]](https://en.wikipedia.org/wiki/Java_(programming_language)#cite_note-FOOTNOTEGoslingJoySteeleBracha20141-15) and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "[write once, run anywhere](https://en.wikipedia.org/wiki/Write_once,_run_anywhere)" (WORA),[[16]](https://en.wikipedia.org/wiki/Java_(programming_language)#cite_note-16) meaning that [compiled](https://en.wikipedia.org/wiki/Compiler) Java code can run on all platforms that support Java without the need for recompilation

**JSP & Servelet:**

Servlet can be described in many ways, depending on the context.

* Servlet is a technology which is used to create a web application.
* Servlet is an API that provides many interfaces and classes including documentation.
* Servlet is an interface that must be implemented for creating any Servlet.
* Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
* Servlet is a web component that is deployed on the server to create a dynamic web page.

**Apacahe Tomcat server:**

**Apache Tomcat**, often referred to as **Tomcat Server**, is an open-source [Java Servlet Container](https://en.wikipedia.org/wiki/Servlet_container) developed by the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation) (ASF). Tomcat implements several [Java EE](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) specifications including [Java Servlet](https://en.wikipedia.org/wiki/Java_Servlet), [JavaServer Pages](https://en.wikipedia.org/wiki/JavaServer_Pages" \o "JavaServer Pages)(JSP), [Java EL](https://en.wikipedia.org/wiki/Unified_Expression_Language), and [WebSocket](https://en.wikipedia.org/wiki/WebSocket), and provides a "pure [Java](https://en.wikipedia.org/wiki/Java_(programming_language))" [HTTP](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) [web server](https://en.wikipedia.org/wiki/Web_server) environment in which [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) code can run.

**Rest APIS**:

REST or [RESTful API](https://www.mulesoft.com/resources/api/restful-api) design (Representational State Transfer) is designed to take advantage of existing protocols. While REST can be used over nearly any protocol, it usually takes advantage of HTTP when used for Web APIs. This means that developers do not need to install libraries or additional software in order to take advantage of a REST API design. REST API Design was defined by Dr. Roy Fielding in his 2000 doctorate dissertation. It is notable for its incredible layer of flexibility. Since data is not tied to methods and resources, REST has the ability to handle multiple types of calls, return different data formats and even change structurally with the correct implementation of hypermedia.

**Log4j:**

Apache **Log4j** is a [Java](https://en.wikipedia.org/wiki/Java_platform)-based [logging](https://en.wikipedia.org/wiki/Computer_data_logging) utility. It was originally written by [Ceki Gülcü](https://en.wikipedia.org/w/index.php?title=Ceki_G%C3%BClc%C3%BC&action=edit&redlink=1" \o "Ceki Gülcü (page does not exist)) and is part of the [Apache Logging Services](https://en.wikipedia.org/w/index.php?title=Apache_Logging_Services&action=edit&redlink=1)project of the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation). Log4j is one of several [Java logging frameworks](https://en.wikipedia.org/wiki/Java_logging_frameworks).Gülcü has since started the [SLF4J](https://en.wikipedia.org/wiki/SLF4J) and Logback[[3]](https://en.wikipedia.org/wiki/Log4j" \l "cite_note-3) projects, with the intention of offering a successor to Log4j.The Apache Log4j team has created a successor to Log4j 1 with version number 2.[[4]](https://en.wikipedia.org/wiki/Log4j#cite_note-4) Log4j 2 was developed with a focus on the problems of Log4j 1.2, 1.3, java.util.logging and Logback, and addresses issues which appeared in those frameworks. In addition, Log4j 2 offers a plugin architecture which makes it more extensible than its predecessor. Log4j 2 is not backwards compatible with 1.x versions,[[5]](https://en.wikipedia.org/wiki/Log4j#cite_note-5) although an "adapter" is available.

IMPACT OF METHOD TRACER ON BUISNESS:

In today’s life the most precious thing in the world is“TIME”.People wanna that their work should be completes in less time,for that they can addopt any type of thing or method which can completes their work as fast as possible.In todays buisness enviourment every buisness company wants that their product is manufactured and provided to their client as soon as possible.In IT industry the most valueable thing is time,that their software processing is as fast as possible and give the correct output that their client wanna.

Imagine that your embedded software project is progressinthis probg nicely through the final stages of testing when, suddenly, the software unexpectedly crashes. The engineers who found this bug cannot give you a reproducible test case, yet they are able to get the product to crash periodically. Sound familiar? As an engineer, you want to solve this problem. Now imagine that there is a tool that gives you nearly perfect visibility into what your software was doing when the system crashed, making it easy to find and fix these intermittent and difficult bugs. The technology that makes this dream a reality is trace data generated by embedded microprocessors, and the tools that collect and analyze this trace data make it possible for you to analyze what went wrong and find the cause of the undesirable behavior.This type of method tracer can find out the bug easily and fasters the speed of processing and decreases the execution time of the program