ASSIGNMENT

FROM: PRIYANKA ELANGOVAN

1. **Write about maven life cycle and its commands?**

Maven is a powerful project management tool that is based on POM (project object model), used for projects build, dependency and documentation. It is a tool that can be used for building and managing any Java-based project.

The default Maven lifecycle and its 8 steps: Validate, Compile, Test, Package, Integration test, Verify, Install and Deploy

1. **Maven Commands:**

* **mvn clean:** Cleans the project and removes all files generated by the previous build.
* **mvn compile:** Compiles source code of the project.
* **mvn test-compile:** Compiles the test source code.
* **mvn test:** Runs tests for the project.
* **mvn package:** Creates JAR or WAR file for the project to convert it into a distributable format.
* **mvn install:** Deploys the packaged JAR/ WAR file to the local repository.
* **mvn deploy:** Copies the packaged JAR/ WAR file to the remote repository after compiling, running tests and building the project.

1. **What are Web Services?**

A web service is any piece of software that makes itself available over the internet and uses a standardized XML messaging system. XML is used to encode all communications to a web service. For example, a client invokes a web service by sending an XML message, then waits for a corresponding XML response. As all communication is in XML, web services are not tied to any one operating system or programming language—Java can talk with Perl; Windows applications can talk with Unix applications.

1. **what is rest controller?**

Rest Controller is a Spring annotation that is used to build REST API in a declarative way. Rest Controller annotation is applied to a class to mark it as a request handler, and Spring will do the building and provide the RESTful web service at runtime.

1. **Web services and its methods:**

* GET method:

The GET method refers to a Hypertext Transfer Protocol (HTTP) method that is applied while requesting information from a particular source. It is also used to get a specific variable derived from a group. The HTTP POST asks for input of information from the supplying browser into the server's message system.

* PUT method:

PUT method is used to update resource available on the server. Typically, it replaces whatever exists at the target URL with something else. You can use it to make a new resource or overwrite an existing one.

* POST method:

POST is a request method supported by HTTP used by the World Wide Web. By design, the POST request method requests that a web server accept the data enclosed in the body of the request message, most likely for storing it. It is often used when uploading a file or when submitting a completed web form.

* DELETE method:

The DELETE method is defined to be idempotent, which means that sending the same HTTP DELETE request multiple times will have the same effect on the server and will not additionally affect the state or cause additional side effects.

1. **GIT life cycle:**

*Git* project have various stages like *Creation, Modification, Refactoring*, and *Deletion* and so on. Irrespective of whether this project is tracked by Git or not, these phases are still prevalent. However, when a project is under Git version control system, they are present in three major Git states in addition to these basic ones. Here are the three Git states:

* *Working directory*
* *Staging area*
* *Git directory*

***Working Directory***

Consider a project residing in your local system. This project may or may not be tracked by Git. In either case, this project directory is called your Working directory.

### *****Staging Area*****

It's important to make a quick note of the term called ***indexing***here. ***Indexing*** is the process of adding files to the staging area. In other words, index constitutes of files added to the staging area. This term will be explained again in the coming tutorial on ***Git terminologies***.

### *****Git Directory*****

***Remote repository means mirror or clone of the local Git repository in GitHub*.**And***pushing means uploading the commits from local Git repository to remote repository hosted in GitHub.***

1. **Git commands:**

**git checkout**

You can use the **checkout**command to switch the branch that you are currently working on.

Syntax: git checkout <branch name>

**git init**

This is the command you need to use if you want to start a new empty repository or to reinitialize an existing one in the project root. It will create a .git directory with its subdirectories. It should look like this:

Syntax: git init <repository name>

**git commit**

This one is probably the most used Git command. After changes are done locally, you can save them by “committing” them. A commit is like local a snapshot of the current state of the branch, to which you can always come back. To create a new commit, type this command in Git Bash:

Syntax: git commit -m "<commit message>"

**git push**

Git push will push the locally committed changes to the remote branch. If the branch is already remotely tracked, simply use it like this (with no parameters):

git push

**git pull**

Using git pull will fetch all the changes from the remote repository and merge any remote changes in the current local branch.

git log

Let’s you explore the previous revisions of a project. It provides several formatting options for displaying committed snapshots.

git rebase

Rebasing lets you move branches around, which helps you avoid unnecessary merge commits. The resulting linear history is often much easier to understand and explore.

git reset

Undoes changes to files in the working directory. Resetting lets you clean up or completely remove changes that have not been pushed to a public repository.

git revert

Undoes a committed snapshot. When you discover a faulty commit, reverting is a safe and easy way to completely remove it from the code base.

git add

Moves changes from the working directory to the staging area. This gives you the opportunity to prepare a snapshot before committing it to the official history.

1. **Difference between centralized and distributed version control**

