**Assignment-15.2**

1. Explain the working and the differences between Maven, Gradle and SBT in detail.

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

Maven is an attempt to apply patterns to a project's build infrastructure in order to promote comprehension and productivity by providing a clear path in the use of best practices. It is similar to Apache Ant or Gradle. It does everything on its own we don’t need any additional tools or scripts incorporating other common tasks like downloading & installing necessary libraries etc. Maven primary goal is to provide developer –

• A comprehensive model for projects which is reusable, maintainable, and easier to comprehend.

• Plugins or tools that interact with this declarative model.

1.Builds 2. Documentation 3.Reporting 4.Dependencies 5.SCMs 6.Releases

Distribution Maven can provide benefits for your build process by employing standard conventions and practices to accelerate your development cycle while at the same time helping you achieve a higher rate of success. Maven’s dependency mechanism help to download all the necessary dependency libraries automatically, and maintain the version upgrade as well. Maven uses Convention over Configuration which means developers are not required to create build process themselves. Developers do not have to mention each and every configuration detail.

Maven provides sensible default behavior for projects. When a Maven project is created, Maven creates default project structure. Developer is only required to place files accordingly and he/she need not to define any configuration in pom.xml. It should be noted that there should be a single POM file for each project. All POM files require the project element and three mandatory fields: groupId, artifactId and version.

**Gradle**: Gradle is an advanced general purpose build management system based on Groovy and Kotlin. Gradle supports the automatic download and configuration of dependencies or other libraries. It supports Maven and Ivy repositories for retrieving these dependencies. This allows reusing the artifacts of existing build systems. A project using Gradle describes its build via a build.gradle file. This file is located in the root folder of the project. The build file for Gradle builds is based on a **Domain Specific Language (DSL).** In this file you can use a combination of declarative and imperative statements. You can also write Groovy or Kotlin code, whenever you need it. Tasks can also be created and extended dynamically at runtime.

Comparison of Maven and Gradle on the basis of features :

**Feature 1**: How easy is the initial learning curve.

1. Maven is XML based tool, XML is very commonly used/known.
2. If existing project are using maven then developers are comfortable with the system.
3. Gradle is DSL based system and need to learn explicitly.
4. Developers need to learn new system from beginning if they are not familier.

**Feature 2**: How fast are different builds with each tool.

Taken a report from zeroturnaround, they have done detail analysis of the speed of builds with both built tools and found maven and gradle are close enough on build timings.

**Feature 3**: How complex is it to create and maintain the build script?

Maven build scripts are xml based, which has predefined structure and only one way to write. So it makes it more standard and less flexible.

Gradle has its own DSL which is introduced by Gradle itself and tightly connected to Gradle internals But its flexible and simple and short.

**Feature 4**: How many plugins exist and how simple is it to customize your own plugins?

Maven is called “plugin execution framework” Hundreds of plugins exist for Maven and you can create your own plugin is simple. Gradle’s architecture is also plugin-based, It’s easy to write plugins But availability of plugin in community may not be ease.

**Feature 5**: How good is the community and documentation for each tool?

Maven is in the market for very long time, Documentation is good, Lots of resources and help available in open community and forum. Gradle is very recent. It is open source but still under control of gradleware. They have option for commercial support.

**Feature 6:** How well does each tool integrate with developer tools? (IDE, App Server, CI server).

With many years of background Maven has full support to almost each tool and every category (IDE, App Server, CI). Lacking in the App Server and CI ServerCategories, mainly due to newness.

One can decide the tool on the basis of their user case and requirement.

Gradle You may not be using other features which is used by google for android development as those features may or may not match with your needs however gradle is essentially emerging as popular tool and you may definitely want to have a look at it. Maven is widely used in most of the companies and have robust defined workflow around it. Developers are very much familiar with the system. However this may not be viable option as android development need to use gradle. Maven and Gradle (Both) This would be the one more option for a company with wide range of development area which may give developers flexibility to adopt tool as per their choice with in given workflow.

**SBT** : Simple Build Tool is a general purpose build tool written in Scala for JVM developers. It borrows good ideas from other successful build tools like Ant, Maven, and Gradle.

Features are :

1.Default project layouts

2.Built-in tasks

3.Plugin architecture

4.Declarative Dependency management

5.Code over Configuration: A DSL for build tool

Apart from the feature set mentioned above sbt also provides the following additional features like-

6.Interactive nature: It isn't just a build tool, it also provides an interactive environment to work in.

7.Scala REPL integration. The maven workflow is hardcoded. SBT has a more flexible way of configuring how your builds should progress. But if you really need this, you may be overcomplicating things.