Task 1

Maven Life Cycle

Follows a structured sequence of build phases:

validate: Checks project structure and configurations.

compile: Converts source code to bytecode.

test: Runs unit tests.

package: Bundles compiled code into a JAR/WAR.

verify: Executes integration tests.

install: Stores build in the local Maven repository.

deploy: Sends the package to a remote repository.

What is pom.xml and Why It's Used

Core configuration file in a Maven project.

Defines project metadata, dependencies, plugins, and build instructions.

Advantages:

Centralized dependency and plugin management.

Simplifies build automation.

Establishes standard project structure.

Integrates tools for building and deploying applications.

How Dependencies Work in Maven

Managed in the <dependencies> section of pom.xml.

Automatically downloads and manages required libraries.

Handles transitive dependencies (dependencies of dependencies).

Example:

xml

CopyEdit

```
<dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-lang3</artifactId>
  <version>3.12.0</version>
</dependency>
```

Types of Maven Repositories

Local Repository: Cached on developer's machine (~/.m2/repository).

Central Repository: Default global source for dependencies.

Remote Repositories: Custom/private repositories used in enterprises.

Use mvn dependency:tree to view dependency hierarchy.

Building Multiple Modules with Maven

Uses a parent pom.xml to manage multiple sub-modules.

Declare modules using:

xml

CopyEdit

```
<modules>
  <module>core</module>
  <module>ui.apps</module>
  <module>ui.content</module>
</modules>
```

Build all modules: mvn clean install

Building a Specific Module

Use: mvn clean install -pl module-name -am

-pl: Specifies the module to build.

-am: Builds required dependent modules as well.

Roles of AEM Modules

ui.apps: Contains configuration, templates, and components.

ui.content: Holds pages, assets, and content packages.

ui.frontend: Contains front-end assets (JS, CSS, React, etc.).

Why Use Run Modes in AEM

Environment-specific configurations for development, staging, production.

Benefits:

Improved performance.

Enhanced security.

Examples:

author mode: For content creation and editing.

publish mode: For serving content to end-users.

What is the Publish Environment

Delivers finalized AEM content to users.

Syncs published content from the author instance to live environment.

Why Use the Dispatcher in AEM

AEM’s caching and load-balancing tool.

Key functions:

Boosts performance by caching.

Reduces load on AEM instances.

Adds a layer of request filtering and security.

Accessing CRX/DE

CRX/DE: Web-based content and configuration manager for AEM.

URL: http://localhost:4502/crx/de/index.jsp

Allows browsing and editing the JCR repository directly.