Prerequisite Context

Before starting these steps, a **WAR-based Java web project** was already created using Maven. The default `index.html` inside the project was **replaced with a custom** `index.html` prior to deployment.

Step 1: System Preparation

Ensure the system packages are up to date and Java (JDK 17) is available for running Tomcat and building the project.

Step 2: Apache Tomcat Setup

Download Apache Tomcat 9 from the official Apache source and extract it on the server. After extraction, the archive is cleaned up to keep the environment tidy.

Step 3: Tomcat User & Manager Configuration

Configure Tomcat users by updating `tomcat-users.xml` to include appropriate roles required for the Tomcat Manager application. This enables authenticated access for deployment and monitoring.

Step 4: Enable Remote Access to Tomcat Manager

Update the Manager application's `context.xml` to relax IP-based restrictions, allowing the Tomcat Manager UI to be accessed remotely.

Step 5: Start Tomcat Server

Start the Tomcat server and verify that the default Tomcat page and Manager application are accessible through the browser on port `8080`.

Step 6: Clone the Application Repository

Clone the previously created WAR-based project (Nexus) from the remote Git repository to the server.

Step 7: Maven Setup & Verification

Install Maven, configure the environment, and verify the installation to ensure the build toolchain is ready.

Step 8: Build the WAR Application

Use Maven to clean and package the project. This process generates a deployable `.war` file inside the project's `target` directory.

Step 9: Prepare Tomcat for Deployment

Stop the Tomcat server to ensure a clean deployment environment before placing the new WAR file.

Step 10: Deploy the WAR File

Copy the generated `Nexus.war` file into Tomcat's `webapps` directory. Tomcat automatically detects WAR files placed here.

Step 11: Restart Tomcat Server

Restart Tomcat to trigger automatic extraction and deployment of the WAR file.

Step 12: Application Verification

Access the deployed application using:

```
http://<server-ip>:8080/Nexus
```

The application should load successfully and serve the **custom** `index.html` that replaced the default one earlier.

Final Outcome

- Java environment configured
- Apache Tomcat installed and secured with Manager access
- Maven-built WAR application deployed successfully

- Application accessible via browser

This approach represents a **manual WAR deployment workflow**, commonly used for learning, demos, and basic production environments.