**Project summary**

**Goal:** stand up a clean, reproducible **Java + Selenium + JUnit 5** automation project with **Maven**, prove it runs both in **Chrome** and **Brave**, and verify a simple assertion (“Google” page title).  
**Why this matters:** it demonstrates you can bootstrap a modern Selenium stack, manage dependencies with Maven, run tests headlessly/locally, and interpret build logs/warnings.

**What we built (structure)**

selenium-demo/

├─ pom.xml <- Maven config (Selenium, JUnit, Surefire)

├─ src/

│ └─ test/

│ └─ java/

│ └─ com/example/

│ ├─ AppTest.java <- Chrome smoke test (title == "Google")

│ └─ BraveTest.java <- Brave smoke test (same check)

└─ target/

└─ surefire-reports/ <- JUnit XML + text reports after mvn test

**Key technologies & decisions**

* **Java 17** as target (modern LTS).
* **Maven** to resolve and pin library versions.
* **Selenium 4.x** using **Selenium Manager**, so you don’t manually download ChromeDriver/BraveDriver.
* **JUnit 5** test framework, executed by **maven-surefire-plugin**.

**The pom.xml (why it’s written this way)**

* Declares versions in <properties> (easier to bump later).
* Adds **selenium-java** and **junit-jupiter** dependencies.
* Configures **Surefire** to run JUnit 5 tests with mvn test.

(You fixed a “Malformed POM” earlier by removing a duplicate <dependencies> tag—classic copy-paste issue. Maven must have **one** <dependencies> block inside <project>.)

**The tests (what they do)**

**AppTest.java (Chrome)**

* Creates a ChromeDriver (Selenium Manager auto-resolves the matching driver).
* get("https://www.google.com")
* Reads driver.getTitle() and asserts it equals "Google".
* driver.quit() to close the session.

**BraveTest.java (Brave)**

* Creates ChromeOptions and points setBinary(...) to Brave’s executable (Brave is Chromium-based, so we still use ChromeDriver).
* Remainder is identical: open Google, assert title, quit.

Note: If Brave is installed in the default Windows path, the binary is typically:

C:\Program Files\BraveSoftware\Brave-Browser\Application\brave.exe

Adjust if yours differs.

**How we ran it (end-to-end flow)**

1. **Rename typo**: AppTest.java.java → AppTest.java so Maven/JUnit sees it.
2. **From project root**: mvn test
3. **What Maven did**
   * Read pom.xml, downloaded dependencies from Maven Central (you saw lots of “Downloading from central…” lines).
   * Compiled tests (target/test-classes).
   * **Surefire** discovered JUnit 5 tests (com.example.\*Test) and executed them.
4. **Selenium Manager** detected your local **Chrome 139** and **Brave 139**, fetched matching drivers, and launched browsers.
5. **Assertions passed** → build finished with:
6. Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
7. BUILD SUCCESS

**Interpreting the warnings you saw**

* **CDP (Chrome DevTools Protocol) warning**
* WARNING: Unable to find CDP implementation matching 139 ...

This is safe to **ignore** unless you’re using Selenium’s **DevTools API** (e.g., network throttling, console logs). Your tests use only WebDriver commands, so it’s fine. If you ever need DevTools features, add the matching selenium-devtools-vXXX artifact for your Chromium version.

* **SLF4J logger warning**
* No SLF4J providers were found. Defaulting to NOP.

Means you didn’t add a logging backend. Optional. Add slf4j-simple if you want pretty logs.

**Where the results live**

* **Console summary** shows green tests.
* **Detailed reports** in target/surefire-reports/:
  + TEST-com.example.AppTest.xml
  + TEST-com.example.BraveTest.xml
  + A plain-text summary file as well.  
    These are CI- and report-tool friendly.