Prerequisites and Installation Guide windows(all framework)

Step 1: Check Python Installation

Before proceeding with Bugasura CLI setup, verify if Python is installed on your Windows system.

1. Open Command Prompt (cmd) or PowerShell
2. Run the following command:
   bash
   ```
   python --version
   ```

3. If Python is installed, you'll see the version number. If not, proceed to Step 2.

Step 2: Install Python (If Not Already Installed)

1. Visit the official Python website: https://www.python.org/downloads/
2. Download the latest Python version for Windows
3. Run the installer with the following important settings:
   - ✅ Check "Add Python to PATH" during installation
   - ✅ Check "Install pip" (usually selected by default)
   - Choose "Install Now" for default installation

Step 3: Configure Environment Variables

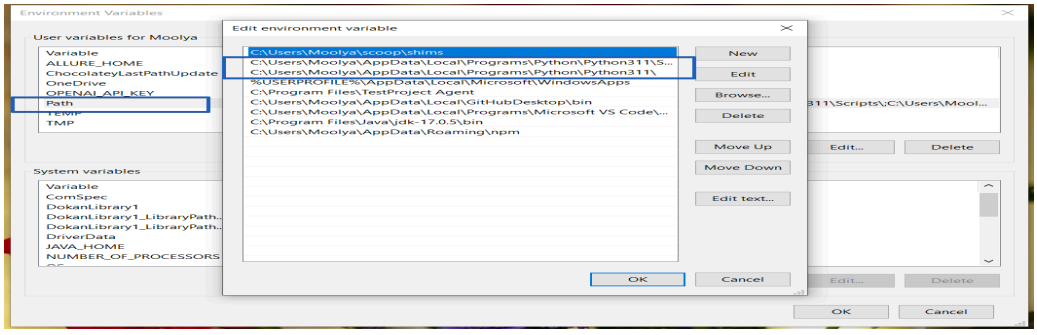
After Python installation, you need to add Python paths to your system's environment variables.

Required Paths to Add:

Based on the screenshot provided, add these paths to your PATH environment variable:

```
C:\Users\Moolya\AppData\Local\Programs\Python\Python311\Scripts\
```

```
C:\Users\Moolya\AppData\Local\Programs\Python\Python311\
```



macOS Setup Instructions

Prerequisites for macOS

macOS comes with Python pre-installed by default. However, you may need to use Python 3 specifically for Bugasura CLI.

Step 1: Verify Python Installation on macOS

1. **Open Terminal** (Applications → Utilities → Terminal)
2. **Check Python 3 installation:**

   bash

   python3 --version

3. **Check pip3 installation:**

   bash

   pip3 --version

> **Note:** macOS typically has Python 2.7 by default, but we need Python 3 for Bugasura CLI.

✅ 1. Playwright

None

```
npx playwright test --reporter=junit
```

Config (playwright.config.ts):

None

```
reporter: [['junit', { outputFile: 'results.xml' }]]

reporter: [['list'], ['junit', { outputFile: 'reports/junit-report.xml' }]]
```

Package.json:

Need to add this for integration

Run : npm test

This will execute the tests and generate report and upload to bugasura

```JavaScript
"scripts": {
    "test": "npx playwright test filename.spec.ts --project=chromium ;
bugasura UPLOAD_RESULTS ./reports/junit-report.xml --api_key=<your api
key> --team_id=<team id> --project_id=<project id> --testrun_id=<testrun
id> --server= <client> ",
    "test:chromium": "npx playwright test --project=chromium"
  }
```

✅ 2. Cypress

```None
npm install --save-dev mocha-junit-reporter
```

cypress.config.js (E2E config):

```JavaScript
reporter: 'mocha-junit-reporter',
reporterOptions: {
  mochaFile: 'results/test-results.[hash].xml',
  toConsole: true
}
```

✅ 3. WebdriverIO

```
None
npm install --save-dev @wdio/junit-reporter
```

wdio.conf.js:

```
JavaScript
reporters: ['spec', ['junit', {
  outputDir: './results',
  outputFileFormat: (opts) => `results-${opts.cid}.xml`
}]]
```

✅ 4. Pytest

```
None
pip install pytest pytest-junitxml
```

Run with:

```
None
pytest --junitxml=results.xml
```

✅ 5. Postman (Newman CLI)

```
None
npm install -g newman newman-reporter-junitfull
```

Run with:

```
None
newman run collection.json -r junitfull
--reporter-junitfull-export results.xml
```

✅ 6. JMeter

- Use built-in JUnit formatter in **JMeter Plugins**.
- CLI:

```
None
jmeter -n -t test.jmx -l results.jtl -e -o report
```

- Convert `.jtl` to JUnit XML via XSLT or use **JMeter Plugins Manager → JUnit Report Listener**.

✅ 7. Selenium (Java)

Use **JUnit/TestNG**, depending on your test framework.

For **JUnit 5**:

```
None
<plugin>
  <groupId>org.junit.platform</groupId>
  <artifactId>junit-platform-surefire-provider</artifactId>
</plugin>
```

<!-- Surefire Plugin for XML reports -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M9</version>

<configuration>

```
            <includes>
               <include>**/TestRunner.java</include>
            </includes>
            <systemPropertyVariables>
               <cucumber.plugin>junit:target/surefire-reports/result.xml</cucumber.plugin>
            </systemPropertyVariables>
         </configuration>
      </plugin>
```

For macOS

Once this is added create new file run.sh

```
#!/bin/bash

echo "✅ Running tests..."
mvn clean test

if [ $? -eq 0 ]; then
 echo "✅ Test execution completed. Uploading results to Bugasura..."
else
 echo "⚠️ Tests finished with some errors. Still attempting upload..."
fi

bugasura UPLOAD_RESULTS target/surefire-reports/result.xml \
 --api_key  \
 --team_id  \
 --project_id  \
```

```
--testrun_id \

--server=<client> if separate instance is created
```

Windows

Add this plugin in pom.xml

<!-- Surefire Plugin for XML reports -->

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>3.0.0-M9</version>
  <configuration>
    <includes>
      <include>**/TestRunner.java</include>
    </includes>
    <systemPropertyVariables>
      <cucumber.plugin>junit:target/surefire-reports/result.xml</cucumber.plugin>
    </systemPropertyVariables>
  </configuration>
</plugin>
```

For windows

Once this is added create new file in the root of project run.bat

```
@echo off
```

```
setlocal


:: Cucumber tag to run
set "TAG_NAME=@YourTestTag"(to run particular test file)


:: Path to test report generated after execution
set "REPORT_PATH=target\surefire-reports\cucumber-results.xml"


:: === BUGASURA CREDENTIALS ===
set "API_KEY=your_bugasura_api_key"
set "TEAM_ID=your_team_id"
set "PROJECT_ID=your_project_id"
set "TESTRUN_ID=your_testrun_id"


:: === OPTIONAL: Add Python Scripts to PATH (only if needed) ===
set "PATH=%PATH%;C:\Path\To\Python\Scripts"


:: === CHECK AND UPLOAD REPORT ===
echo Checking if test report exists...
if not exist "%REPORT_PATH%" (
   echo ❌ Report not found at: %REPORT_PATH%
   goto end
)


echo ✅ Report found. Uploading to Bugasura...
bugasura UPLOAD_RESULTS "%REPORT_PATH%" ^
```

```
--api_key %API_KEY% ^
--team_id %TEAM_ID% ^
--project_id %PROJECT_ID% ^
--testrun_id %TESTRUN_ID%


:: === RESULT LOGGING ===
if %ERRORLEVEL% neq 0 (
   echo ❌ Upload to Bugasura failed with exit code %ERRORLEVEL%
) else (
   echo ✅ Upload to Bugasura successful!
)


:end
endlocal
```

JUnit XML is auto-generated by Maven Surefire or Gradle.

✅ 8. TestNG

Maven Surefire Plugin already outputs `testng-results.xml`.

To customize:

```
None
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
```

```
  <version>3.1.2</version>
</plugin>
```

✅ 9. Robot Framework

```
None
robot --output output.xml tests/
```

- The default `output.xml` can be converted to JUnit:

```
None
robot --xunit results.xml output.xml
```
