WebdriverIO Login Automation Framework: Complete Guide

1. Project Setup: WebdriverIO Automation

1.1. Initialize Node.js Project

```
npm init -y
```

1.2. Install WebdriverIO and Dependencies

```
npm install --save-dev @wdio/cli @wdio/local-runner @wdio/mocha-framework @wdio/spec-reporter webdriverio chromedriver
```

1.3. Configure WebdriverIO

```
npx wdio config -y
```

- This generates `wdio.conf.js` and a sample test structure.

1.4. Project Structure

```
Cursor-Automation-Javascript/
├── test/
│   ├── pageobjects/
│   │   ├── login.page.js
│   │   ├── page.js
│   │   └── secure.page.js
│   └── specs/
│       └── test.e2e.js
├── wdio.conf.js
├── package.json
└── README.md
```

---

2. Writing Login Test Scenarios

2.1. Valid Login Test

- Uses correct credentials (`tomsmith` / `SuperSecretPassword!`)
- Checks for a success alert

2.2. Invalid Login Test

- Uses wrong credentials
- Checks for an error alert

2.3. Example Test (test/specs/test.e2e.js)

```
it('should login with valid credentials', async () => {
    await LoginPage.open()
    await LoginPage.login('tomsmith', 'SuperSecretPassword!')
    await expect(SecurePage.flashAlert).toBeExisting()
    await expect(SecurePage.flashAlert).toHaveText(
        expect.stringContaining('You logged into a secure area!'))
})

it('should not login with invalid credentials', async () => {
    await LoginPage.open()
    await LoginPage.login('wronguser', 'wrongpass')
    await expect(SecurePage.flashAlert).toBeExisting()
    await expect(SecurePage.flashAlert).toHaveText(
        expect.stringContaining('Your username is invalid!'))
})
```

---

3. Allure Reporting Integration

3.1. Install Allure Reporter

```
npm install --save-dev @wdio/allure-reporter allure-commandline
```

3.2. Update `wdio.conf.js`

```
reporters: [
  'spec',
  ['allure', { outputDir: 'allure-results', disableWebdriverStepsReporting: true, disableWebdriverScreenshotsReporting: false }]
],
```

3.3. Generate and View Allure Report Locally

```
npm run wdio
npx allure generate allure-results --clean -o allure-report
npx allure open allure-report
```

---

4. GitHub Integration

4.1. Create a GitHub Repository

- Go to https://github.com/new
- Create a new repo (do not initialize with README)

4.2. Push Your Code

```
git init
git add .
git commit -m "Initial commit: WebdriverIO login automation framework"
git remote add origin https://github.com/YOUR-USERNAME/YOUR-REPO-NAME.git
git branch -M main
git push -u origin main
```

---

5. Jenkins CI/CD Integration

5.1. Add Jenkinsfile to Project Root

Example:

```
pipeline {
    agent any

    stages {
        stage('Install') {
            steps { sh 'npm install' }
        }
        stage('Test') {
            steps { sh 'npm run wdio' }
        }
        stage('Allure Report') {
            steps { sh 'npx allure generate allure-results --clean -o allure-report' }
        }
    }
    post {
        always {
            allure includeProperties: false, jdk: '', results: [[path: 'allure-results']]
        }
    }
}
```

5.2. Create Jenkins Pipeline Job

- Go to Jenkins Dashboard → New Item → Pipeline
- Set "Pipeline script from SCM" and provide your GitHub repo URL

5.3. Install Allure Jenkins Plugin

- Jenkins Dashboard → Manage Jenkins → Manage Plugins → Available → Search "Allure Jenkins Plugin" → Install

5.4. View Allure Report in Jenkins

- After a successful build, click the **Allure Report** link on the build page.

---

6. Troubleshooting & Common Issues

6.1. `npm` Not Recognized

- Ensure Node.js and npm are installed and in your system PATH.
- Restart your terminal after installation.

6.2. Jenkins Cannot Find Chrome/Chromedriver

- Make sure Chrome and Chromedriver are installed on your Jenkins agent.
- For Linux, you may need to install with `apt-get install google-chrome-stable`.

6.3. Allure Report Not Showing in Jenkins

- Ensure the Allure plugin is installed.
- Make sure your pipeline includes the `allure` step in the `post` section.
- The `allure-results` directory must be generated by your tests.

---

7. Example README.md

A full README was created and added to your project, including setup, usage, and reporting instructions.

---

8. Summary of Your Queries & Solutions

- **How to set up WebdriverIO for login automation?**
  → Step-by-step setup, test writing, and configuration provided.
- **How to add invalid login scenarios?**
  → Example test added.
- **How to integrate Allure reporting?**
  → Dependencies, config, and Jenkins integration explained.
- **How to push to GitHub?**
  → Git commands and repo setup detailed.
- **How to set up Jenkins CI/CD?**
  → Jenkinsfile, job creation, and Allure plugin steps provided.
- **How to view Allure reports in Jenkins?**
  → Plugin installation and pipeline configuration explained.
- **How to troubleshoot common issues?**
  → Solutions for npm, Chrome, and Allure problems included.

If you need this as a downloadable document (Markdown or Word), or want further customization, let me know!