**JavaScript For Testing**

Duration: 24 Hours

**JavaScript For Testers:**

* Variables and Data Types
* let, const, and var
* Numbers, strings, booleans, null, undefined
* Objects and arrays
* Type conversion and coercion

**Git**

* Start with the main branch
* Create a new-branch
* Update, add, commit, and push changes
* Push feature branch to remote
* Resolve feedback
  + Merge conflict overview
  + Performing merge conflict with IDE support
  + Merge your pull request

Overview of Merge Vs Rebase

**Hands On Labs/ Activities:**

* Implementing the concepts with the initial project structure.
* Setting up the GIT repository for the training project, configure permissions. Implementing the feature branch workflow for participants.

**JavaScript For Testers:**

* Functions and Scope
* Function declarations and expressions
* Arrow functions
* Parameters and return values
* Closures and lexical scope
* this keyword and context

**Hands On Labs/ Activities:**

* Implementing the concepts and pushing code to GIT.

# NodeJS

* An Introduction To The Npm Package Manager Npm Install Packages mechanisms?
* Execute A Package Installed Using Npm
* The Package.Json / Package-Lock.Json mechanisms
* Npm Global Or Local Packages
* The Npx Node.Js Package Runner basics
* Run Node.Js Scripts From The Command Line

**Hands On Labs/ Activities:**

* Implementing the concepts and pushing code to GIT.

**JavaScript For Testers:**

* Control Flow
* Conditional statements (if/else, switch)
* Loops (for, while, do-while)
* try/catch error handling
* Break and continue statements

**Hands On Labs/ Activities:**

* Implementing the concepts and pushing code to GIT.

**JavaScript For Testers:**

Modern JavaScript Features

* Promises and Async Programming
* Creating and consuming promises
* async/await syntax
* Error handling with try/catch
* Promise chaining
* Promise.all() and Promise.race()

**Hands On Labs/ Activities:**

Implementing the concepts and pushing code to GIT.

**JavaScrip For Testers:**

ES6+ Features

* Template literals
* Destructuring assignments
* Spread/rest operators
* Default parameters
* Optional chaining
* Nullish coalescing

DOM Manipulation (for Web Testing)

* Selectors and Elements
* querySelector and querySelectorAll
* Getting and setting attributes
* Modifying element content
* Working with classes and styles

**JavaScrip For Testers:**

DOM Manipulation (for Web Testing)

* Selectors and Elements
* querySelector and querySelectorAll
* Getting and setting attributes
* Modifying element content
* Working with classes and styles
* Events and Event Handling
* Common event types
* Event listeners
* Event bubbling and capturing
* Event delegation

**Hands On Labs/ Activities:**

* Implementing the concepts and pushing code to GIT.

**Automation Testing with Cypress**

# Duration: 40 Hours

**Automation Architecture Best Practices:**

* SOLID Principles
* Clean Architecture Principles
* The Generic Test Automation Architecture Introduction to the Demo App:
  + A demo application to demonstrate real-world usage of automation testing methods, patterns, and workflows.

**Hands On Labs/ Activities:**

* Implementing the GTAA for the sample application.

**Cypress Automation:**

# Cypress

* App install and overview
* Installing cypress and writing your first test
* How to test forms and custom cypress commands
* How to test multiple pages
* How to test user journeys
* How to write a test
* Cypress runs tests in the browser

**Cypress Studio:**

* Key Advantages of Cypress Studio
* Configure Cypress Studio
* Record the steps

**Cypress Basics**

* Command chaining
* The asynchronous nature of cypress
* Waiting and retry-ability

**Hands On Labs:**

# Cypress Basics

* How to debug failing tests
* Cypress JavaScript concepts
* Custom commands & plugins - extends cypress functionality using javascript modules
* Aliases & variables - cypress-specific approach to storing and retrieving test data
* Cypress command queue - understanding how commands are enqueued and executed
* Fixture handling - loading and working with external test data

**Hands On Labs:**

# ypress Advanced Topics

* Running in continuous integration (ci)
* intercepting http requests
* api and integration tests
* building custom commands
* viewport and browser testing
* cypress methods
* using data for dynamic tests

**Cypress Limitations**: (Handling Multiple Tabs/ Windows)

* No native support for multiple browsers/ tabs.
  + Discussion on workarounds with some approaches
  + Using Cy.origin
  + Modifying target attributes
  + Stubbing requests / response
  + Using the cypress-puppeteer plugin.

**Hands On Labs:**

**Cucumber for JavaScript:**

* Gherkin: writing your stories in a user-friendly language
* Features
* Scenarios
* Steps - Given, When, Then, And, But
* Data Tables in Steps
* Creating Multiple Features and Step files
* Organizing features and scenarios with tagging
* Data-Driven Scenario Outlines
* Usage of Background

**Hands On Labs/ Activities:**

* Implementing the concepts and pushing code to GIT.
* Writing feature file using the Gherkin syntax for common scenarios

**Cucumber for JavaScript:**

* Adding/Deleting New Steps
* Step Definitions Binding
* Running and Debugging Scenarios and step Definitions
* Parameterization to Promote step Definition Reusability
* Working with hooks
* Usage of Context Injection

**Hands On Labs/ Activities:**

* Based of Java Petstore application, Automation Practice Heroku App

**Automation Testing Playwright**

**Duration: / 40 Hours**

**Automation Architecture Best Practices:**

* SOLID Principles
* Clean Architecture Principles
* The Generic Test Automation Architecture Introduction to the Demo App:
  + A demo application to demonstrate real-world usage of automation testing methods, patterns, and workflows.

**Hands On Labs/ Activities:**

* Implementing the GTAA for the sample application.

**PLAYWRIGHT**:

Introduction to Playwright

* Understand Playwright Architecture and components
* Playwright Installation & project Setup
* Install Node.js and Visual studio code
* Generate Package.json and get Playwright Dependencies
* Introduction to the framework design
* The essential playwright objects that make up a test Playwright Test Runner and command line features
* The test object ,
* Expect object
* Fixtures available for use from the Test runner.

**Hands On Labs/ Activities:**

* Based of Java Petstore application, Automation Practice Heroku App

**PLAYWRIGHT**:

Tools:

* Playwright Inspector
* Playwright CodeGen
* Playwright UI- Mode Execution and TimeLine views
* Playwright CLI options to execute scripts
  + 1. Run all the tests
  + 2. Run a single test file
  + 3. Run a set of test files
  + 4. Run tests in headed browsers
  + 5. Run tests in a particular browser (config-less mode)
  + 6. Run tests in all browsers (config-less mode)
  + 7. Run in debug mode with Playwright Inspector
* Recording your testing sessions using “codegen” feature

**Hands On Labs/ Activities:**

* Based of Java Petstore application, Automation Practice Heroku App

**Cucumber for JavaScript:**

* Gherkin: writing your stories in a user-friendly language
* Features
* Scenarios
* Steps - Given, When, Then, And, But
* Data Tables in Steps
* Creating Multiple Features and Step files
* Organizing features and scenarios with tagging
* Data-Driven Scenario Outlines
* Usage of Background

**Hands On Labs/ Activities:**

* Implementing the concepts and pushing code to GIT.
* Writing feature file using the Gherkin syntax for common scenarios

**PLAYWRIGHT**:

* Playwright locator strategies and how to construct them
* Basic Assertion in writing the tests with Playwright
* Playwright Configuration

**Performance Considerations:**

* Parallel Test Execution
* Use of Fixtures
* Network Interception
* Headless Mode
* Performance Metrics
* Stable Selectors
* Focused and Isolated Tests

**Cucumber for JavaScript:**

* Adding/Deleting New Steps
* Step Definitions Binding
* Running and Debugging Scenarios and step Definitions
* Parameterization to Promote step Definition Reusability
* Working with hooks
* Usage of Context Injection

**Hands On Labs/ Activities:**

* Based of Java Petstore application, Automation Practice Heroku App

**PLAYWRIGHT:**

Introduction to Playwright Commands and its Asynchronous Nature

* Web applications to Practice Playwright Automation
* Understanding different commands with Playwright
* Grabbing the text for validations using Playwright text command Playwright Asynchronous nature and its promise handling using async
* Handling Async promises with Playwright
* Handling Web Controls UI using Playwright
* How to verify and automate check boxes with Playwright

**Hands On Labs/ Activities:**

Based of Java Petstore application, Automation Practice Heroku App and The Cypress Real world App

**PLAYWRIGHT:**

* Handling static dropdowns using select command with Playwright
* Typing in Test Fields
* Click action
* Checkbox and Radio button action
* Element State
* Mouse Events
* Handling Web Tables.
* Advance Automation to handling Alerts,popups,Child Windows using Playwright
* How Playwright auto handles Alerts in web Apps
* Handling Child tabs with Playwright
* Navigating browser controls using Playwright
* Handling Mouse Event

**Hands On Labs/ Activities:**

* Based of Java Petstore application, Automation Practice Heroku App and The Cypress Real world App

**Automation Architecture Best Practices:**

* Adapting application Types in the GTAA GUI, API , Databases etc
* Using multiple application types within the same framework
* Calling API Tests inside test cases

**Hands On Labs/ Activities:**

* Based of Java Petstore application, Automation Practice Heroku App and The Cypress Real world App