

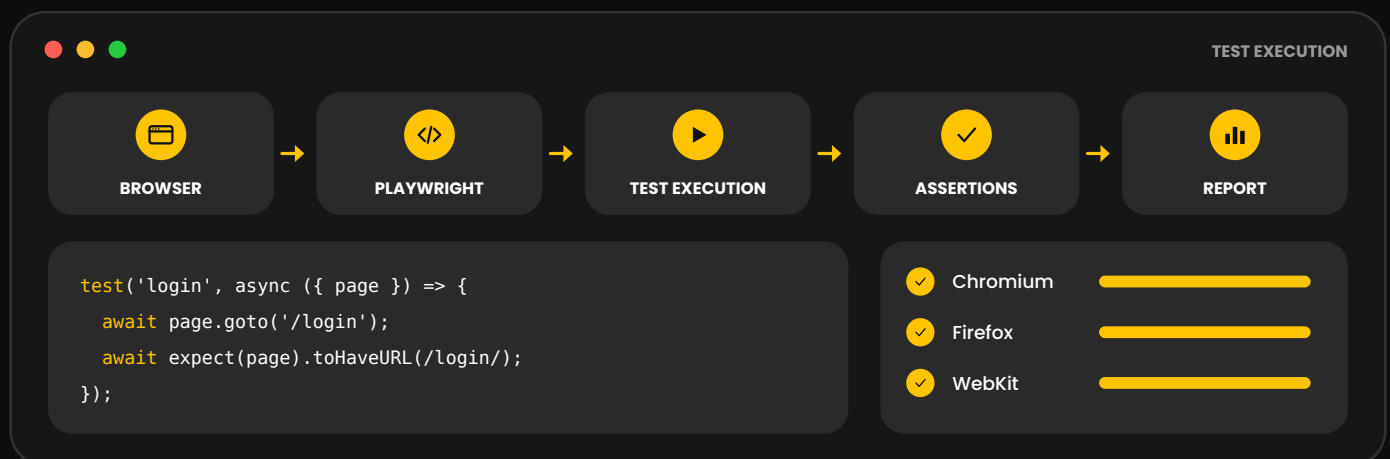
CAREER-FOCUSED PROGRAM

PLAYWRIGHT WITH TYPESCRIPT

Complete Course Content

Web Automation Testing – From Fundamentals to Framework Design

Master web automation testing with Playwright and TypeScript, from fundamentals to framework design, API testing, AI-assisted testing and an end-to-end capstone project.



TRAINER

Premchand

DURATION

45 Days

LEVEL

Beginner to Job-ready

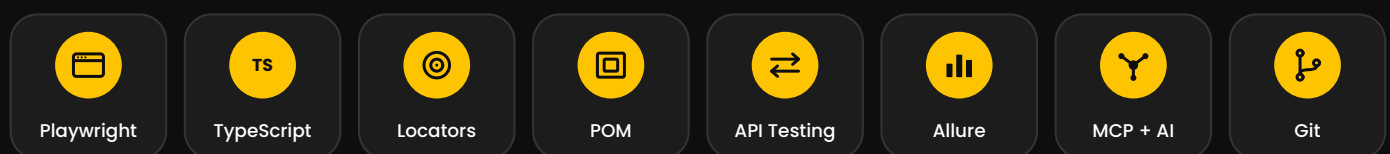
TOOLS

Node.js, TypeScript, VS Code, Playwright Test, Allure, Git

STRUCTURE

Part A: Playwright Automation
Part B: TypeScript Fundamentals
Part C: Capstone, Interview & Resume

WHAT YOU WILL MASTER



ENROLL TODAY

Start your automation testing career with RaiseTech

☎ 9390271992

8247329739

📍 Flat No. 308, Nilgiri Block, Aditya Enclave, Ameerpet, Hyderabad – 500016

OVERVIEW

From Fundamentals to Framework Design

This 45-day program takes you from the fundamentals of web automation testing to designing a complete test framework. You will learn Playwright with TypeScript – locators, assertions, web elements, Page Object Model, data-driven testing, reports, REST API testing and AI-assisted testing with Playwright MCP – and put everything together in an end-to-end e-commerce capstone project.

45

Days

19

Playwright Topics

25

TypeScript Topics

1

Capstone Project

PROGRAM SNAPSHOT



Who the course is for

Learners at any starting point: beginner to job-ready, with no prior coding required.



Learning approach

Daily practice assignments after each module, hands-on exercises, an MCP exercise, a capstone project and mock interviews.



Duration

45 days of structured training, from Playwright fundamentals to a complete capstone framework.



Level

Beginner to Job-ready. No prior coding required – TypeScript is taught from the basics in Part B.



Tools

Node.js, TypeScript, VS Code, Playwright Test, Allure and Git.



Course structure

Part A: Playwright Automation. Part B: TypeScript Fundamentals. Part C: Capstone, Interview & Resume.

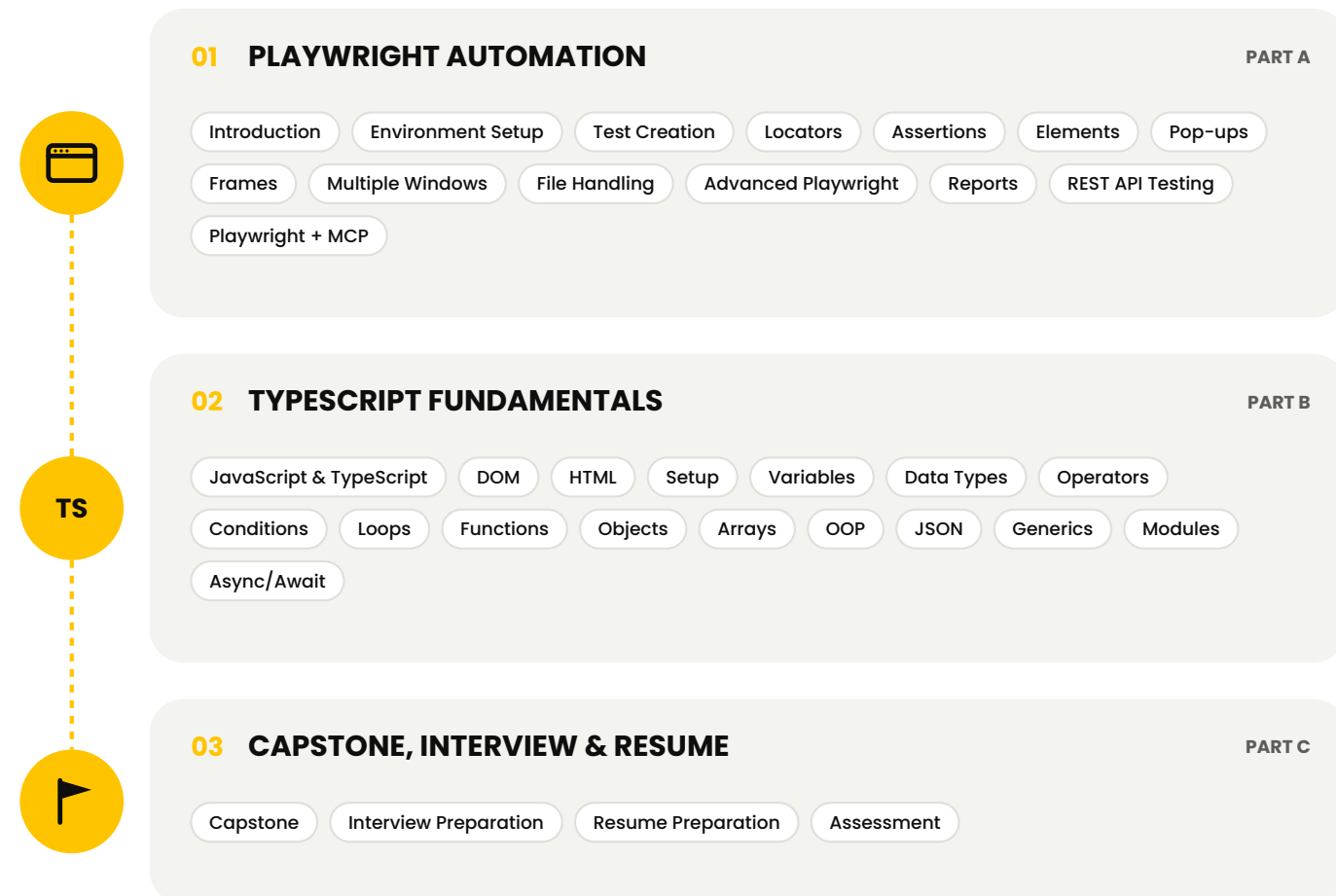


CAREER OUTCOME

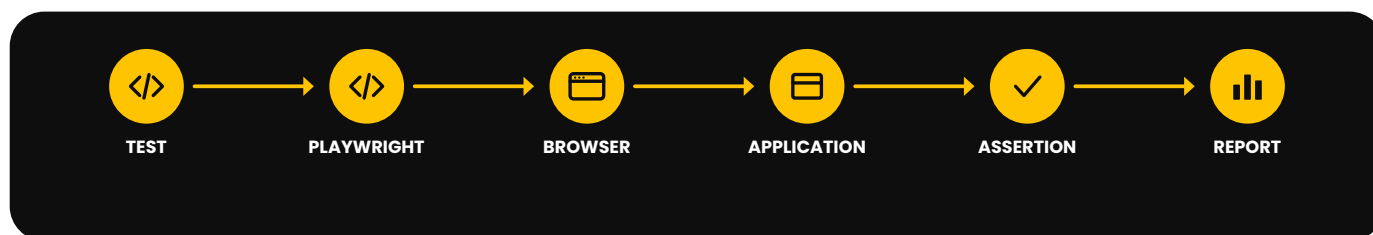
Become job-ready: interview-oriented sessions, mock interviews, automation resume preparation, Naukri / LinkedIn profile tips and a framework project with a README that can go on a resume.

WHAT YOU WILL LEARN

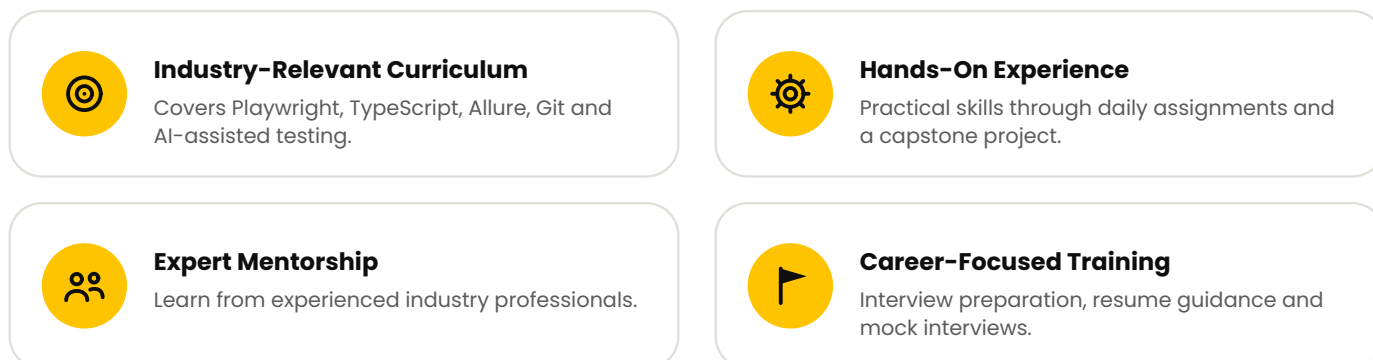
Your Learning Path



PLAYWRIGHT AUTOMATION FLOW



WHY RAISETECH



WHAT YOU WILL LEARN

Curriculum at a Glance



MODULE 01

Introduction to Playwright

What is Playwright · Features & Architecture · Playwright vs Selenium vs Cypress · Chromium, Firefox, WebKit · Auto-waiting



MODULE 02

Installation & Environment Setup

npm init playwright@latest · Project Structure · playwright.config.ts · Browsers · VS Code Extension



MODULE 03

Creating & Running Tests

First Test · test & expect · Terminal & Editor Runs · page Fixture · goto, Title & URL Checks



MODULE 04

Locators & Element Identification

XPath · CSS Selectors · Built-in Locators · Filtering & Chaining · Stable Locators



MODULE 05

Playwright Test Generator

Codegen · Recording Tests · Cleaning & Refactoring Generated Code



MODULE 06

Configuration, Execution & Assertions

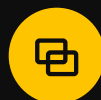
Timeouts · Retries · Workers · Projects · Headed vs Headless · Parallel Runs · Hard & Soft Assertions · Web-first Assertions



MODULE 07

Working with Web Elements

Textboxes · Buttons · Checkboxes · Radio Buttons · Dropdowns · Auto-suggest & Static Dropdowns



MODULE 08

Pop-ups, Frames & Multiple Windows

Alerts, Confirms & Prompts · Dialog Event · frameLocator · Nested Frames · New Tabs & Windows



MODULE 09

Calendar, Mouse, Keyboard & File Handling

Date Pickers · Hover · Right & Double Click · Drag and Drop · Key Presses · Uploads & Downloads



MODULE 10

Advanced Playwright Concepts

Hooks · test.describe · Screenshots · Trace Viewer · Tags · Annotations · Page Object Model · Data-driven Testing



MODULE 11

Reports & Test Management

HTML, List & JSON Reporters · Allure Reports · Steps & Attachments



MODULE 12

REST API Testing

APIRequestContext · GET, POST, PUT, DELETE · Headers & Authentication · Response Validation · API + UI



MODULE 13

Playwright + MCP

Model Context Protocol · Playwright MCP · Accessibility Snapshots · Browser Tools · Test Agents · Best Practices



MODULE 14

TypeScript Fundamentals

JavaScript & TypeScript · DOM · HTML · Setup · Variables & Data Types · Operators · Conditions · Loops



MODULE 15

TypeScript Functions, Objects & Arrays

Functions · Objects · Arrays & Methods · Date · Strings & Numbers · Regex · Exception Handling



MODULE 16

TypeScript OOP, JSON & Generics

Classes · Static · Encapsulation · Inheritance · Prototype · Polymorphism · JSON · Generics · Type Narrowing



MODULE 17

Modules, Promises & Async/Await

export & import · Default vs Named Exports · Promises · async/await



MODULE 18

Capstone, Interview & Resume

E-commerce Capstone · Interview Session · Mock Interviews · Resume Preparation · Naukri / LinkedIn Tips



PART A

Playwright Automation

19 topics: from introduction and setup to reports, REST API testing and Playwright + MCP.



PART B

TypeScript Fundamentals

25 topics: from JavaScript and TypeScript basics to OOP, JSON, generics and async/await.



PART C

Capstone, Interview & Resume

Capstone project, interview-oriented session, resume preparation and assessment.

HOW IT ALL FITS TOGETHER





PART A

Playwright Automation

Modules 01 – 13



MODULE 01

Introduction to Playwright

Playwright Overview

- ▶ What is Playwright? Features and architecture
- ▶ Playwright vs Selenium vs Cypress

Browsers & Reliability

- ▶ Supported browsers (Chromium, Firefox, WebKit) and languages
- ▶ Auto-waiting and why tests are less flaky



MODULE 02

Installation & Environment Setup

Project Setup

- ▶ npm init playwright@latest
- ▶ Project structure: playwright.config.ts, tests folder, package.json

Browsers & Editor

- ▶ Installing browsers
- ▶ Installing the VS Code Playwright extension

TOOLS

Node.js, npm, VS Code, Playwright Test



MODULE 03

Creating & Running Tests

Your First Test

- ▶ Writing the first test with test and expect
- ▶ page fixture, goto, title and URL checks

Running Tests

- ▶ Running from the terminal
- ▶ Running from the editor



MODULE 04

Locators & Element Identification

XPath

- ▶ Absolute and relative XPath
- ▶ contains(), text()
- ▶ Axes

CSS Selectors

- ▶ id, class and attribute selectors
- ▶ Child and descendant selectors
- ▶ nth-child

Choosing Stable Locators

- ▶ Choosing stable locators

Built-in Locators

- ▶ getByRole
- ▶ getByText
- ▶ getByLabel
- ▶ getByPlaceholder
- ▶ getByAltText
- ▶ getByTitle
- ▶ getByTestId

Filtering, Chaining & Multiple Matches

- ▶ Filtering and chaining locators
- ▶ Handling multiple matches: first, last, nth, all



MODULE 05

Playwright Test Generator

Codegen

- ▶ Using the Codegen tool to record tests
- ▶ Cleaning and refactoring generated code



MODULE 06

Configuration, Execution & Assertions

playwright.config.ts

- ▶ Timeouts
- ▶ Retries
- ▶ Workers
- ▶ Projects

Test Execution

- ▶ Headed vs headless
- ▶ Browser selection
- ▶ Running a single file or test
- ▶ Parallel execution
- ▶ test.only / test.skip

Hard & Soft Assertions

- ▶ Hard assertions with expect
- ▶ Soft assertions with expect.soft

Web-first Assertions

- ▶ toBeVisible, toHaveText, toHaveValue
- ▶ toHaveURL, toBeChecked, toBeEnabled
- ▶ Custom timeouts and negative assertions



MODULE 07

Working with Web Elements

Textboxes & Buttons

- ▶ Editbox / TextBox: fill, type, clear
- ▶ Push buttons: click

Checkboxes & Radio Buttons

- ▶ Checkboxes: check, uncheck, isChecked
- ▶ Radio buttons

Dropdowns

- ▶ Dropdowns: selectOption
- ▶ Auto-suggest dropdowns
- ▶ Static dropdowns



MODULE 08

Pop-ups, Frames & Multiple Windows

Pop-ups

- ▶ JavaScript alerts, confirms and prompts: accept and dismiss
- ▶ Handling the dialog event
- ▶ Browser authentication pop-ups

Frames

- ▶ frameLocator and frame()
- ▶ Nested frames

Multiple Windows

- ▶ Browser contexts and multiple pages
- ▶ Handling new tabs and windows with waitForEvent('popup')
- ▶ Switching between pages



MODULE 09

Calendar, Mouse, Keyboard & File Handling

Calendar / Date Pickers

- ▶ Selecting dates through the UI
- ▶ Dynamic date logic with the Date object

Mouse Actions

- ▶ Mouse hover
- ▶ Right click
- ▶ Double click
- ▶ Drag and drop

Keyboard Actions

- ▶ Key-based operations: press, keyboard.type, shortcuts

File Uploads & Downloads

- ▶ setInputFiles for single and multiple files
- ▶ Uploads without an input element (file chooser)
- ▶ File download handling



MODULE 10

Advanced Playwright Concepts

Hooks & Grouping

- ▶ Playwright hooks: beforeAll, afterAll, beforeEach, afterEach
- ▶ Grouping tests with test.describe

Screenshots & Trace Viewer

- ▶ Capture screenshots (page, element, full page, on failure)
- ▶ Trace Viewer: recording and analysing traces

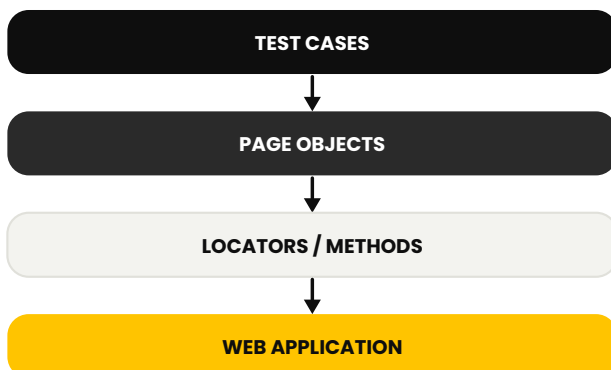
Tags & Annotations

- ▶ Tagging tests and running by tag (--grep)
- ▶ Playwright annotations: skip, fixme, fail, slow, tag

Page Object Model (POM)

- ▶ Page classes, constructors, reusable methods, fixtures

PAGE OBJECT MODEL



Data-driven Testing

- ▶ Data-driven testing with JSON/CSV
- ▶ Environment variables



MODULE 11

Reports & Test Management

Built-in Reporters

- ▶ HTML, list and JSON reporters

Allure Reports

- ▶ Setup
- ▶ Steps and attachments
- ▶ Generating and opening the report

TOOLS

Allure Report



MODULE 12

REST API Testing

Requests

- ▶ Playwright request context (APIRequestContext)
- ▶ GET, POST, PUT, DELETE requests
- ▶ Headers, authentication and request body

Validation & UI

- ▶ Response validation
- ▶ Combining API setup with UI verification

API TESTING FLOW



API REQUEST

01



AUTHENTICATION

02



RESPONSE

03



VALIDATION

04



UI VERIFICATION

05



MODULE 13

PLAYWRIGHT + MCP

AI-ASSISTED TESTING

An MCP server that lets an AI assistant control a real browser: explore an application in plain English, generate Playwright TypeScript tests, then review and refactor them.



What is MCP

- ▶ MCP (Model Context Protocol): clients, servers, tools and why it matters for testing

What is Playwright MCP

- ▶ An MCP server that lets an AI assistant control a real browser
- ▶ Accessibility snapshots vs screenshots: how the AI 'sees' the page without a vision model

Setup

- ▶ Running the server with `npx @playwright/mcp@latest`
- ▶ Registering it in VS Code / GitHub Copilot, Claude, Cursor or other MCP clients
- ▶ Server options: browser choice, headless mode, viewport, storage state and persistent profiles

Core Browser Tools

navigate snapshot click type select hover
 drag file upload handle dialogs tabs
 screenshots network console messages

Hands-on: AI-Assisted Test Creation

- ▶ Exploring an application in plain English and having the AI perform flows step by step
- ▶ Generating Playwright TypeScript tests from MCP sessions
- ▶ Reviewing and refactoring them into POM

Playwright Test Agents

- ▶ Planner, generator and healer concepts for creating and repairing tests

Planner Generator Healer

Best Practices

- ▶ Prompt design
- ▶ Stable locators
- ▶ Keeping humans in the loop
- ▶ Validating AI-generated assertions

Security & Limits

- ▶ Trusting only known sites
- ▶ Avoiding real credentials in prompts
- ▶ Flaky or non-deterministic output

MINI EXERCISE

Explore a demo site with MCP, produce a test, and run it in CI-style headless mode.

TS

PART B

TYPESCRIPT FUNDAMENTALS

Modules 14 – 17

Learn JavaScript and TypeScript from the ground up – the foundation of every Playwright test.



TS

MODULE 14

TypeScript Fundamentals

Introduction to JavaScript and TypeScript

- ▶ What is JavaScript? Features, advantages, role in web development
- ▶ What is TypeScript and why Playwright recommends it
- ▶ Compile-time type checking vs runtime behaviour
- ▶ How TypeScript compiles to JavaScript

What is the DOM (Document Object Model)?

- ▶ Understanding the DOM tree structure
- ▶ How to write and execute a script
- ▶ Accessing and manipulating DOM elements

Introduction to HTML Tags

- ▶ Basic structure of an HTML page
- ▶ Common elements: input, button, select, table, iframe, a
- ▶ Integrating scripts with HTML

Environment Setup

- ▶ Installing Node.js and npm
- ▶ Installing VS Code and the Live Server extension
- ▶ Installing TypeScript, running with tsc and ts-node, tsconfig.json basics
- ▶ Comments and statements

Variables and Data Types

- ▶ Declaring variables: var, let, const
- ▶ Primitive types: string, number, boolean, null, undefined, bigint, symbol

Variables and Data Types (contd.)

- ▶ Type annotations and type inference
- ▶ any, unknown, void, never
- ▶ Arrays, tuples, enums, union and literal types, type aliases

Operators

- ▶ Arithmetic, assignment, comparison, logical and ternary operators
- ▶ Optional chaining (?.) and nullish coalescing (??)

Conditional Statements

- ▶ if, if...else, else if ladder
- ▶ switch case

Loops and Jumping Statements

- ▶ while, for, do...while, for...of, for...in
- ▶ break and continue

{ }

MODULE 15

TypeScript Functions, Objects & Arrays

Functions

- ▶ Function declaration and expression
- ▶ Arrow functions
- ▶ Typed parameters and return types, optional, default and rest parameters
- ▶ Function overloads, callbacks

Working with Objects

- ▶ Creating and accessing objects
- ▶ Object properties and methods
- ▶ Interfaces and type aliases for object shapes
- ▶ Destructuring and spread operator

Arrays

- ▶ Creating and iterating arrays
- ▶ Common array operations
- ▶ Typed arrays and readonly arrays

Array Methods

- ▶ `map()`, `filter()`, `reduce()`, `forEach()`, `find()`
- ▶ `some()`, `every()`, `sort()`, `slice()`, `splice()`

Date Constructor

- ▶ Working with the Date object

MINI PROJECT

Digital Clock

Strings and Numbers

- ▶ String manipulation and template literals
- ▶ Number methods and conversions
- ▶ Regular expression basics (useful for assertions)

Exception Handling

- ▶ `try`, `catch`, `finally`, `throw`
- ▶ Custom errors and handling runtime errors gracefully



MODULE 16

TypeScript OOP, JSON & Generics

OOP Concepts

- ▶ Classes and objects
- ▶ Methods and constructors
- ▶ Access modifiers: `public`, `private`, `protected`; `readonly`

Static Keyword

- ▶ Static methods and properties

Encapsulation

- ▶ Data hiding using private fields and closures
- ▶ Getters and setters

Inheritance

- ▶ Class inheritance, method overriding, the `super` keyword
- ▶ Abstract classes

Prototype

- ▶ Prototype and prototype chaining
- ▶ Adding methods dynamically

Polymorphism

- ▶ Method overriding and dynamic method behaviour
- ▶ Implementing interfaces

Working with JSON

- ▶ `JSON.parse`: JSON text to object
- ▶ `JSON.stringify`: object to JSON text
- ▶ Reading test data from JSON files

Generics and Type Narrowing

- ▶ Generic functions, interfaces and classes
- ▶ `typeof`, `instanceof` and custom type guards



MODULE 17

Modules, Promises & Async/Await

Modules

- ▶ `export` and `import`, default vs named exports
- ▶ Organizing code across files

Promises and async/await

- ▶ Synchronous vs asynchronous code
- ▶ Promises, `then/catch`
- ▶ `async` functions and `await` - the foundation of every Playwright test



PART C · MODULE 18

CAPSTONE PROJECT

End-to-End E-commerce Automation

Automate an end-to-end e-commerce flow: login, search, cart, checkout - built as a complete framework you can showcase.



FRAMEWORK COMPONENTS



Page Object Model



Fixtures



Test Data



Tags



Retries



Allure Reporting



Trace on Failure



Git Repository



README



WHAT YOU BUILD

- ▶ Framework with POM, fixtures, test data, tags and retries
- ▶ Allure reporting and trace on failure
- ▶ Git repository and a README that can go on a resume

PART C · MODULE 18

Career Preparation



INTERVIEW-ORIENTED SESSION

Interview preparation

- ▶ Playwright interview questions
- ▶ TypeScript interview questions
- ▶ Locator strategies
- ▶ Auto-waiting
- ▶ Flaky test handling
- ▶ POM design
- ▶ Framework architecture
- ▶ CI/CD basics
- ▶ Scenario-based questions
- ▶ Coding questions
- ▶ Mock interviews



RESUME PREPARATION

Automation resume

- ▶ Structuring an automation resume
- ▶ Describing the framework and project experience
- ▶ Project experience presentation
- ▶ Naukri profile tips
- ▶ LinkedIn profile tips



MOCK INTERVIEWS

Scenario-based and coding questions, mock interviews and a final mock interview to finish the course.

ASSESSMENT & PRACTICE



01

Daily Practice

Assignments after each module



02

Mid-Course Test

At the end of Part B



03

Capstone Review

Review of the e-commerce capstone



04

Final Mock Interview

Interview practice to finish the course



05

MCP Exercise

AI-generated test reviewed and refactored by the student

TECH STACK

Tools & Technology



Language & Runtime

Node.js (LTS) npm TypeScript



Editor

Visual Studio Code
Playwright Test extension



Test Automation

Playwright Test Chromium Firefox
WebKit



AI-Assisted Testing

Playwright MCP server GitHub Copilot
Claude Cursor



Reporting

Allure Report



Version Control

Git GitHub

HOW THE TOOLS WORK TOGETHER



PUT SKILLS TO WORK

Projects & Career Outcomes

FEATURED PROJECT



CAPSTONE PROJECT

End-to-End E-commerce Automation

Automate an end-to-end e-commerce flow: login, search, cart, checkout.

Page Object Model

Fixtures

Test data

Tags

Retries

Allure

Trace on failure

Git

SKILLS YOU WILL GAIN

● Playwright Automation

● Web Automation Testing

● TypeScript

● Test Design

● Locator Strategies

● Assertions

● Page Object Model

● Test Framework Design

● Data-driven Testing

● API Testing

● Allure Reporting

● Trace Analysis

● MCP-based Testing

● AI-Assisted Testing

● CI/CD Fundamentals

● Test Debugging

● E-commerce Automation



AI IN YOUR WORKFLOW

Go beyond scripted tests. With Playwright MCP, an AI assistant can explore your application in plain English and generate Playwright tests - which you then review, validate and refactor into POM, keeping humans in the loop.

Planner

Generator

Healer



ROLES THIS PREPARES YOU FOR

- ▶ Playwright Automation Tester
- ▶ QA Automation Engineer
- ▶ Test Automation Engineer
- ▶ SDET
- ▶ Software Test Engineer
- ▶ Automation QA Engineer



LEARN TODAY, LEAD TOMORROW

We bridge the gap between learning and real-world application with hands-on, practical training. Join RaiseTech and start building your career in automation testing with Playwright and TypeScript.



CALL US

9390271992



CALL US

8247329739



VISIT US

Flat No. 308, Nilgiri Block, Aditya Enclave, Ameerpet, Hyderabad - 500016

Ready to start your automation testing career?

Call or visit RaiseTech, Ameerpet to know more about batches and fees.

