PALYWRIGHT+ SQL + API TESTING+ AI TESTING

1.Playwright Automation Syllabus

(TS + JS + GenAI + Agentic AI)

Typescript:

🧩 **Module 1: Introduction to TypeScript**

Topics:

- What is TypeScript?
- Why TypeScript over JavaScript
- TypeScript vs JavaScript
- Installing Node.js & TypeScript
- Setting up project using VS Code
- Understanding tsconfig.json

Practical:

- Setup a basic TypeScript project

Outcome:

Students understand **why TypeScript is used in automation**

⚙️ **Module 2: Basic Syntax & Data Types**

Topics:

- Variables (let, const)
- Data Types:
    - number
    - string
    - boolean
    - any
    - unknown
    - void

- Arrays and Tuples
- Enums

Practical:

- Write simple scripts using different data types

Outcome:

Students learn **core building blocks**

🔁 Module 3: Operators & Control Statements

Topics:

- Arithmetic operators
- Comparison operators
- Logical operators
- Conditional statements:
  - if / else
  - switch
- Loops:
  - for
  - while
  - do-while

Outcome:

Students understand **decision-making logic**

🔧 Module 4: Functions

Topics:

- Function declaration
- Arrow functions
- Optional & default parameters
- Return types

Practical:

- Write reusable functions for test logic

Outcome:

Students learn **code reusability**

🧱 Module 5: Objects & Interfaces

Topics:

- Object creation
- Type alias
- Interfaces
- Readonly properties

Practical:

- Create objects for test data

Outcome:

Students understand **structured data modeling**

🧩 Module 6: Classes & OOP Concepts

Topics:

- Classes & Objects
- Constructor
- Access modifiers:
    - public
    - private
    - protected
- Inheritance
- Polymorphism
- Abstraction

Practical:

- Create Page Object Model classes

Outcome:

Students learn **OOP for automation frameworks**

📦 Module 7: Modules & Imports

Topics:

- Export & Import
- Default export
- Named export
- File structure for automation projects

Outcome:

Students understand **code organization**

🔄 Module 8: Asynchronous Programming

Topics:

- Callbacks
- Promises
- async/await
- Error handling (try/catch)

Practical:

- Handle API responses in tests

Outcome:

Students understand **async behavior in automation**

JavaScript:

🧩 Module 1: Introduction to JavaScript

Topics:

- What is JavaScript?
- Role of JavaScript in Testing
- Setting up environment (Node.js, VS Code)
- Running JS files using Node

Practical:

- Run first JS program

Outcome:

Students understand **how JS is used in QA automation**

⚙️ Module 2: Variables & Data Types

Topics:

- Variables: var, let, const
- Data Types:
  - string
  - number
  - boolean
  - undefined
  - null
- Type checking (typeof)

Practical:

- Store test data (username, password)

Outcome:

Students learn **basic data handling**

🔁 Module 3: Operators & Conditions

Topics:

- Arithmetic operators
- Comparison operators (== vs ===)
- Logical operators (&&, ||)
- Conditional statements:
  - if / else
  - switch

Practical:

- Validate login conditions

Outcome:

Students understand **decision-making logic**

🔄 Module 4: Loops & Iterations

Topics:

- for loop
- while loop
- for...of
- forEach

Practical:

- Iterate test data sets

Outcome:

Students can **handle multiple test inputs**

🔧 Module 5: Functions (Core for QA)

Topics:

- Function declaration
- Function expressions
- Arrow functions
- Parameters & return values

Practical:

- Create reusable functions (login, validation)

Outcome:

Students learn **code reusability in tests**

🧱 Module 6: Arrays (Very Important)

Topics:

- Creating arrays
- Array methods:
  - push, pop
  - shift, unshift
  - map
  - filter
  - find

Practical:

- Handle multiple test data
- Filter valid/invalid users

Outcome:

Students can **manage test data efficiently**

📘 **Playwright Automation Syllabus (TS + JS + GenAI + Agentic AI)**

🎯 **Course Objective**

To enable QA engineers to **build scalable automation frameworks using Playwright with JavaScript & TypeScript**, and leverage **Gen AI & Agentic AI** to improve productivity, test design, and maintenance.

🧩 Introduction to Automation & Playwright

Topics:

- What is Automation Testing?
- Why Playwright?
- Playwright vs Selenium vs Cypress
- Supported languages (JS, TS, Python, C#)
- Architecture of Playwright

Practical:

- Install Playwright (JS & TS projects)

Outcome:

Students understand **why Playwright is used in modern QA**

⚙️ Module 2: JavaScript & TypeScript Essentials (QA Recap)

Topics:

- JS basics (functions, arrays, objects)
- TS basics (types, interfaces)
- Async/await (very important)

Practical:

- Write reusable test utilities

Outcome:

Students get **programming foundation for automation**

🏗️ Module 3: Playwright Project Setup

Topics:

- Installing Playwright:
  - JS setup
  - TS setup
- Folder structure
- Config file (playwright.config.ts/js)
- Running tests

Practical:

- Setup project from scratch

Outcome:

Students can **start automation project**

🧪 Module 4: Writing First Test

Topics:

- Test structure
- Test syntax (JS & TS)
- Running single & multiple tests
- Test annotations

Practical:

- Automate login test

Outcome:

Students can **write basic test scripts**

🎯 Module 5: Locators & Selectors

Topics:

- Types of locators:
  - getByRole
  - getByText
  - CSS selectors
  - XPath
- Best practices for locators

Practical:

- Identify elements on UI

Outcome:

Students can **identify stable locators**

🔄 Module 6: Actions & Assertions

Topics:

- Actions:
  - click, fill, type
  - select dropdown
  - hover
- Assertions:
  - toBeVisible
  - toHaveText
  - toHaveURL

Practical:

- Validate UI elements

Outcome:

Students can **perform UI validation**

⚡ Module 7: Auto-Waiting & Synchronization

Topics:

- Auto-wait concept
- Explicit waits
- Handling dynamic elements

Outcome:

Students can **handle flaky tests**

🧱 Module 8: Page Object Model (POM)

Topics:

- What is POM?
- Creating page classes
- Reusability

Practical:

- Build login page class

Outcome:

Students learn **framework structure**

📊 Module 9: Test Data Management

Topics:

- Using JSON files
- Parameterization
- Data-driven testing

Practical:

- Run tests with multiple datasets

Outcome:

Students can **handle test data efficiently**

🌐 Module 10: API Testing with Playwright

Topics:

- APIRequestContext
- GET, POST, PUT, DELETE
- Validating responses

Practical:

- API validation scripts

Outcome:

Students can **perform API testing**

🔐 Module 11: Authentication Handling

Topics:

- Login sessions
- Storage state
- Reusing authentication

Outcome:

Students can **avoid repeated logins**

📸 Module 12: Debugging & Reporting

Topics:

- Trace viewer
- Screenshots & videos
- HTML reports

Practical:

- Debug failed test

Outcome:

Students can **analyze failures**

🔁 Module 13: Parallel Execution & Cross Browser Testing

Topics:

- Running tests in parallel
- Browser support:
  - Chromium
  - Firefox
  - WebKit

Outcome:

Students can **optimize execution**

🏗 Module 14: Framework Design (Advanced)

Topics:

- Folder structure (real-time)
- Hooks (before/after)
- Utilities & helpers
- Environment configs

Outcome:

Students can **build scalable framework**

🔄 Module 15: CI/CD Integration

Topics:

- Running Playwright in pipeline

- YAML basics

Tools:

- Azure DevOps
- GitHub Actions

Outcome:

Students can **run tests in CI/CD**

🤖 Module 16: Introduction to Generative AI (Gen AI)

Topics:

- What is Generative AI?
- How Gen AI helps QA
- Prompt engineering basics

Tools:

- ChatGPT
- GitHub Copilot

Outcome:

Students understand **AI fundamentals**

🤖 Module 17: Gen AI in Playwright Automation

Topics:

- Generate test cases using AI
- Generate Playwright scripts (JS/TS)
- Convert manual test cases → automation
- Generate test data
- Debugging with AI

Practical:

- Use AI to create test scripts

Outcome:

Students can **boost productivity using AI**

🤖 Module 18: Introduction to Agentic AI

Topics:

- What is Agentic AI?
- Difference:
    - Traditional automation vs AI agents
- Autonomous testing concepts

Outcome:

Students understand **next-gen automation**

🤖 Module 19: Agentic AI for Testing

Topics:

- Self-healing tests
- AI agents generating tests dynamically
- Autonomous regression execution
- Smart test prioritization

Tools/Concepts:

- AI agents integrated with testing tools
- Workflow automation

Outcome:
Students learn **future of QA automation**

🤖 Module 20: AI + Playwright Integration (Advanced)

Topics:

- AI-driven locator generation
- AI-based test maintenance
- Using AI APIs in Playwright
- Smart assertions

Practical:

- Build AI-assisted test flow

Outcome:
Students can **combine AI with automation**

🧠 Bonus Module: Interview Preparation

Topics:

- Playwright interview questions
- Scenario-based questions
- Debugging questions

📘 Advanced Playwright Framework Syllabus

(JS + TS + API + DB + GenAI + Agentic AI)

🎯 Final Goal

Students should be able to:

- Build **UI + API + DB integrated automation framework**
- Design **scalable architecture**
- Implement **POM + Utilities + Helpers**
- Validate **end-to-end workflows (UI ↔ API ↔ DB)**

- Use **AI for smart automation**

🧩 Module 1: Framework Architecture (Recap)

Topics:

- Hybrid Framework design
- Folder structure:

tests/
pages/
components/
utils/
helpers/
fixtures/
api/
db/
test-data/
config/
reports/

Outcome:

Students understand **complete framework structure**

🏗 Module 2: Core Framework Setup

Topics:

- Setup using Playwright (JS + TS)
- Config management
- Environment setup (dev/qa/prod)

🧱 Module 3: POM + Components (Advanced)

(Already covered – reinforce here)

- Classic POM
- Component-based POM
- Fluent POM
- Hybrid POM

🧰 Module 4: Utilities & Helpers (Recap + Enhancement)

Utilities:

- Random data generator
- Date utilities
- API utility wrapper
- DB connection utility

Helpers:

- Login helper (UI + API)
- Order creation helper
- End-to-end flow helper

🌐 Module 5: API Testing with Playwright (Core)

Topics:

- What is API Testing?
- HTTP Methods:
    - GET
    - POST
    - PUT
    - DELETE
- Status code validation
- Headers & authentication
- Request & Response structure

Playwright API:

- APIRequestContext
- Reusable API client

Example:

```
const response = await request.post('/login', {
  data: { username, password }
});
```

Practical:

- Login API test
- Create user API test

Outcome:

Students can **write API automation scripts**

🔁 Module 6: API Framework Design (Advanced)

Topics:

- API Layer structure:

```
api/
  clients/
  endpoints/
  payloads/
```

Concepts:

- API client wrapper
- Reusable request methods
- Centralized endpoints
- Dynamic payload builders

Outcome:

Students can **design API framework inside Playwright**

🔗 Module 7: API + UI Integration (Real-Time)

Topics:

- Create test data via API
- Validate via UI
- Capture tokens from API → use in UI

Example Flow:

1. Create user via API
2. Login via UI
3. Validate dashboard

Outcome:

Students learn **smart hybrid testing**

🗄 Module 8: Database Testing Basics

Topics:

- Why Database Testing?
- Types:
  - Data validation
  - Data integrity
- SQL Basics:
  - SELECT
  - INSERT
  - UPDATE
  - DELETE

Outcome:

Students understand **DB validation concepts**

🗄 Module 9: Database Integration (Node.js)

Tools:

- MySQL
- PostgreSQL

Topics:

- Connecting DB using Node.js
- Query execution
- Fetching results

Example:

```
const result = await db.query("SELECT * FROM users WHERE id=1");
```

Outcome:

Students can **connect automation with DB**

🔄 Module 10: DB + API + UI Validation (End-to-End)

Real-Time Flow:

1. Create order via API
2. Verify in DB
3. Validate in UI

Validation Points:

- Data consistency
- API vs DB comparison
- UI vs DB comparison

Outcome:

Students become **full-stack QA engineers**

📊 Module 11: Test Data Strategy (Advanced)

Topics:

- Dynamic data creation (API + DB)
- Data cleanup after execution
- Test isolation

Outcome:

Students manage **real-world data problems**

🧪 Module 12: Fixtures for API & DB

Topics:

- API fixtures (auth token)
- DB fixtures (connection setup)
- Sharing context across tests

Outcome:

Students write **clean setup logic**

📸 Module 13: Logging & Debugging (Advanced)

Topics:

- API logs
- DB query logs
- UI logs
- Centralized logging system

🔐 Module 14: Environment & Secrets Management

Topics:

- API keys
- DB credentials
- Secure storage

🤖 Module 15: Gen AI in API & DB Testing

Tools:

- ChatGPT
- GitHub Copilot

Use Cases:

- Generate API test cases
- Generate payloads
- Generate SQL queries
- Debug API failures

Outcome:

Students improve **speed & accuracy**

🤖 Module 16: Agentic AI in End-to-End Testing

Topics:

- AI agents monitoring:
  - API failures
  - DB mismatches
- Auto-healing test flows
- Smart test prioritization

Outcome:

Students learn **next-gen automation**

🏗 Module 17: Complete Framework Implementation

Must Include:

- UI (POM)
- API Layer
- DB Layer
- Utilities
- Helpers
- Fixtures
- Reporting