1. Getting Started with Playwright TypeScript

6. API Automation - API Mocking + GET, POST, PUT, PATCH & DELETE

6.1. POST API Request using Static JSON Object

6.2. POST API Request using Dynamic JSON Object

6.3. POST API Request using TypeSafety Approach

6.4. GET API Request

6.5. Query Parameter

6.6. PUT API Request

6.7. PATCH API Request

6.8. DELETE API Request

6.9. Mock API Requests

6.10. Mock API Response

6.11. Mocking with HAR files - Recording, Modifying & Replaying

7. Playwright with CI CD Tools (Jenkins, Azure DevOps Pipeline & GitHub Actions)

7.1 How to Integrate Playwright with Jenkins + GitHub

- How to Create GitHub Repository & Push Code?
- Generate PAT (Personal Access Token) for GitHub?
- How to Download & Setup Jenkins?
- Install Plugins:
  - NodeJS Plugin, Git Plugin, JUnit Plugin & Allure Plugin.
- How to Add GitHub PAT token in Jenkins?
- How to Create Jenkins Job & Run Playwright Tests?
- How to Attach Artifacts (playwright-report)

- How to Publish Junit Test Results in Jenkins?
- How to Publish Allure Report in Jenkins?

7.2 How to Integrate Playwright with Azure DevOps Pipeline

- Download & Install GIT - https://git-scm.com/downloads
- How to Install Azure Repos extension in VS Code
- How to Create New Repository in Azure DevOps?
- How to Push Source Code from VS Code to Azure DevOps?
- How to Create .yml (YAML) file?
- How to Create Azure DevOps Pipeline?
- How to Run Playwright Tests using ADO pipeline?
- How to Publish JUnit Test Results in ADO?
- How to Attach “playwright-report” in ADO as Artifacts?
- How to Update Test Case Status Automatically in Test Plan (Pipeline + Local)?

7.3 Integrate Playwright with GitHub Actions:

- How to Create .yml (YAML) File?
- How to Run Playwright Tests using GitHub Actions?
    - Workflow - Auto Trigger
    - Workflow - Manual Trigger
- How to Attach “playwright-report“ as Artifacts

*** Topic wise breakdown tutorials ***

1. Getting Started with Playwright & TypeScript

1.1 What is Playwright and Advantages, Limitations

Playwright is an open-source automation testing tool which is used test end to end modern web mobile applications in headed or headless mode.

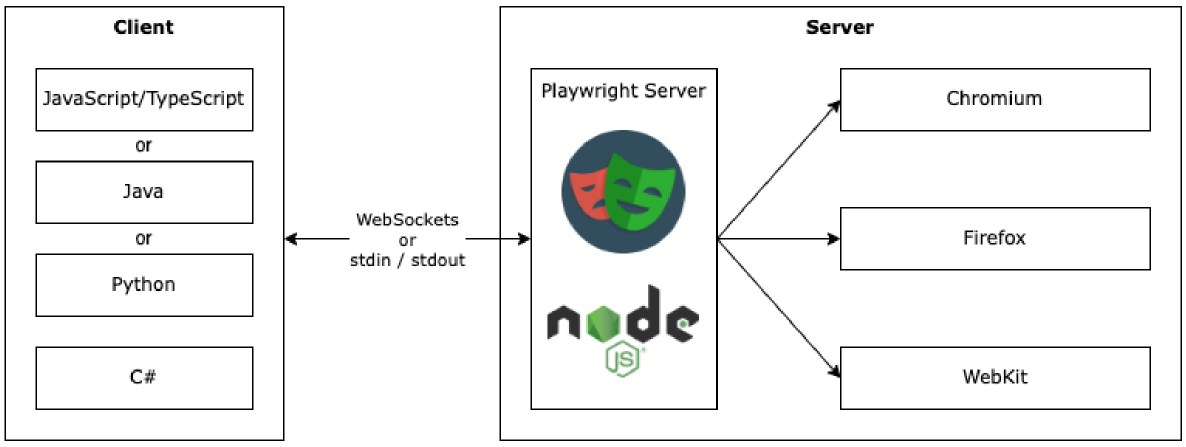
Advantages:

- Cross Browser Testing - Chrome, Edge, Chromium, Firefox & Webkit.
- Cross Platform Testing - Windows, Linux & MacOS.
- Cross Programming Language - TypeScript, JavaScript, Python, .NET & Java.
- Test Mobile Web - Mobile emulation of Google Chrome for Android and Mobile Safari.
- Auto Wait - Example: Click action
- Trace Viewer
- Reporting
- Parallel Execution
- No trade-off & No limit - Cross origin domain support
- Dynamic Wait Assertions
- Faster & Reliable
- State Saving - Example: Application Login
- Powerful Tooling – Codegen, Playwright Inspector & Trace Viewer
- No flaky test

Limitations/Disadvantages:

- Resource Consumption: As it runs on real browser instances This can be a problematic when parallel execution is happening with a system with limited resources
- Playwright does not support for desktop application automation.
- It does support for legacy browsers such Internet Explorer.
- Playwright does not have a big community compared to selenium WebDriver.

1.2 Playwright Architecture



- Client: In the client side our automation test code will be written in different languages such as typescript, JavaScript, python, c# .net etc.
- Server: The server communicates between client and web browser engines such as chrome, Firefox, edge etc.

  Playwright uses CDP to communicate with chromium or chrome browser, similarly playwright has implemented CDP to communicate with other browser such as Firefox webkit etc.
- WebSocket Protocol: WebSocket connection is established to the server from client by using process called Web Socket Handshake, WebSocket sends response as soon as it gets request in real time. Connection will be not terminated until unless connection is closed by either client or server.

- CDP (Chrome DevTools Protocol):
- Client & Server Communication: When test is triggered Client converts automation test into JSON format and sends to server over WebSocket Protocol
  - Playwright communicates all the request over single web-socket protocol connection.
  - Test failure & test flakiness is less as because executing all the tests over single web-socket connection.

1.3 Playwright Vs Cypress vs Selenium WebDriver

Feature	Playwright	Cypress	Selenium
Setup Complexity	Very Easy with JS/TS	Simple and fast setup (JS-based)	High (configuration and dependencies)
Language Support	TypeScript/JavaScript, Python, C#, Java	JavaScript/TypeScript only	Java, Python, JavaScript, C#, Ruby, and more
Speed	Very fast (headless support, parallel execution)	Fast (runs inside browser)	Slower (WebDriver protocol overhead)
Browser Support	Chromium, Chrome, Edge, Firefox, WebKit (Safari)	Chrome, Edge, Electron	All major browsers (Chrome, Firefox, Edge, Safari, IE)
Parallel Execution	Yes	No	No
Debugging	Excellent (built-in tools, browser contexts)	Excellent (real-time feedback, snapshots)	No support
Cross-browser	Yes	Limited (no Firefox/Safari support)	Yes
Community Support	Growing rapidly	Very strong in the JavaScript community	Large, mature community

1.4 Software Requirement -

a) NodeJS

b) Visual Studio Code (VS Code)

c) TypeScript

d) Operating System - Window/MAC/Linux

e) GIT/GitHub

f) Azure DevOps/GitHub Actions

g) Postman Tool for manual API Testing Flow

h) Android Studio for Mobile App Testing

1.5 Install NodeJS & VS Code

https://youtu.be/TkdZymnGZhs?si=LJUa3ka-SZKrx-mP

a) NodeJS – Download & Install

b) Visual Studio Code (VS Code) - Download & Install

1.6 Install Playwright using VS Code

https://youtu.be/-kccujLmsDs?si=K5eJneZZM-vYcQ6Q

Step1: We need to install “Playwright Test for VSCode” extension.

Step2: Press Ctrl+Shift+P or Command+Shift+P, enter “install playwright”

Step3: Select Playwright configurations & click on “OK”

Then wait for some time to complete the installation.

1.7 Playwright Default Folder Structure

https://youtu.be/F48g0-LGAos?si=Gkh3f-Bw6bJDI6JE

1.8 Test Explorer:

https://youtu.be/RSo4RrdW4Mg?si=s8_QIOg-z8gP9IOg

- Run, Debug Tests, Filter

1.9 Setup Playwright Test Automation using CMD or Terminal

https://youtu.be/_KTzU4Gf1_Q?si=UQo-BrklmJd3wUZx

- Install:

Command: npm init playwright@latest

- Choose between TypeScript or JavaScript (default is TypeScript)
- Name of your Tests folder (default is tests folder)
- Add a GitHub Actions workflow to easily run tests on CI
- Install Playwright browsers (default is true)

- Run Test

- Playwright HTML Test Report

1.10 Record Test in VS Code

- Manual Test Scenario

- Record Test

- Run Test

- Playwright HTML Test Report

https://youtu.be/LxDrCdnVhQo?si=z2YDP_R0Rq8nABPH

1.12 Run Test on Google Chrome, Microsoft Edge & Firefox Browsers

https://youtu.be/8C_fphAZ_Ic?si=T5RCIPNSqyPzMTe8

1.12 Generate Readable Playwright HTML Test Report

https://youtu.be/v0Yvwx7phuo?si=OsdEf1_0h2yvaliC

1.14 Commonly used Terminologies in Playwright Testing

https://youtu.be/r57xT1kfl8A?si=TS4_MLfeay73MFOq

- async & await, test, page, expect

- What is async & await? - async is a function, await is valid inside async function only.
  - Zero or more await expressions can be written inside async function
  - When each time async functions is called, which will return resolved promised value or rejected with an exception details.

- What is test?

  The test function is used to define an individual test. It provides a way to structure and organize your tests by giving them a name and it takes 2 arguments.

  - First argument is test case title

  - Second argument is async()

- What is page?

  It is a browser page object, it allows you to interact with web browser & perform actions like click, typing into input fields etc.

Also, it provides access to various methods to interact with web browser.

- What is expect?

  Playwright's expect is powerful and provides a rich set of matchers for various types of checks, such as checking element visibility, text content, existence etc.

  - It lets you write better assertions for end-to-end testing.

1.15 First Playwright Automation Test

1.16 How to use Pick Locator

- Manual Test Scenario
- Automate Test Scenario
- How to use Pick Locator to identify element
- Run Playwright Test
- Playwright Test report

https://youtu.be/9io46lyZlCc?si=MEijgDoyUXiFhpCp

1.17 Record at Cursor

https://youtu.be/letbosZsGo8?si=X2hxdL7Dylaa3HZ6

1.18 How to Run Specific Spec File?

https://youtu.be/rb5C4-3w_7c?si=ec53KZtWo8WXzE0B

1.19 How to Run Playwright Test in Headless mode?

https://youtu.be/QNzw0xrWhPY?si=eu1LCeFD3-71hdq0

1.20 How to Run Playwright Test in Headed mode?

https://youtu.be/Ushdk7FKRqI?si=UNig-3UGIwmq6kH6

1.21 How to Run Playwright Test on Different Browsers using Terminal/CMD

https://youtu.be/zqnD6T03fv0?si=cACXDNJq3YFZTwi0

1.22 Codegen – Record and Play Test

- Manual Test Scenario
- Using Codegen
- Create Spec File
- Run Test
- Playwright Test Report

https://youtu.be/8WfwUHhAAlo?si=liSHIb6FY2Qvpets

Chapter- 02

2.1 How to Capture Screenshots in Playwright?

- Element screenshot
- Page screenshot
- Full Page screenshot

https://youtu.be/0i0U9MSxJpc?si=wpv3pbrv5lqmAgpE

2.2 How to Add Screenshots to Playwright HTML Test Report

- When Test is Failed
- When Test is Passed

https://youtu.be/jF-hneihfJE?si=yjdIEF1FjaRPM6R9

2.3 Locators Strategy in Playwright Test Automation

https://youtu.be/cuZbPdmYTz0?si=ChPpowOB9NWCP58M

a) By Role

Example - await page.getByRole('link',{name: 'value'}).click()

b) By Label

Example - await page.getByLabel('value',{exact : true}).fill('value');

c) By Alt Attribute - For images

Example - await page.getByAltText('alt attribute value').click();

d) By Test Id - For Custom Attributes

Example - await page.getByTestId(‘value).fill(‘testers talk’);

e) By Text

Example - await page.getByText(‘any text’).click(); //partial

Example - await page.getByText(‘any text’, {exact :true}).click();

f) By Placeholder

Example - await page.getByPlaceholder(‘value]).click();

g) XPath

Example - await page.locator(“xpath=//*[@attr=’value’]”).click();

Example - await page.locator(“//*[@attr=’value’]”).click();

h) CSS Selector

Example - await page.locator (“css=tagname[@attr=’value’]”).click();

Example - await page.locator (“tagname[@attr=’value’]”).click();

i) By Title

Example - await page.getByTitle(‘playwright by testers talk).click();

2.4 How to Implement Hooks in Playwright?

i) beforeAll()

ii) beforeEach()

iii) afterEach()

iv) afterAll()

https://youtu.be/URfl51R6XyU?si=ygwoF1sKoRvHFD4G

2.5 Handling Dropdown List

- Manual Test Scenario
- Selecting Dropdowns Value using Value
- Selecting Dropdowns using Visible Text
- Validate Dropdown options

['Jan', 'Feb', 'Mar','Apr', 'May', 'Jun', 'Jul','Aug', 'Sep', 'Oct', 'Nov', 'Dec']

https://youtu.be/dlP-4N_Vd2k?si=SSFp5tx0-jkEWfMh

2.6 How to Switch into IFrames in Playwright?

- How to Drag and Drop Elements in Playwright?

https://youtu.be/pwUxspaA4kY?si=yIMZyqlgOwx6U_nu

2.7 How to perform Mouse Actions in Playwright?

- Mouse Left/Middle/Right Button Clicks
- Mouse Hover
- Double Click

https://youtu.be/7WkWQYrmb0k?si=NybEl-H-F71EsSKv

2.8 How to perform Keyboard Actions in Playwright?

- How to Press 'Enter', 'Tab', 'Delete' & 'Control+A' or 'Meta+A' etc.

https://youtu.be/RWT91MHrYhI?si=HyphRdzskaff_QKZ

2.9 How to Handle Date Picker in Playwright?

- Hardcode Date
- Dynamically Selecting Date
- Selecting Past Date
- Selecting Future Date

https://youtu.be/J9MyZaxZ29Y?si=1ONEJeKy1MLM6oqO

2.10 Assertions in Playwright Test Automation

https://youtu.be/hVsOvwpXfnY?si=MMCUlP8E25mkjoUX

https://youtu.be/jQOeqM7UrCQ?si=bIICD6DCfYNBOL-y

There are 2 types of assertions

1) Hard Assertion

2) Soft Assertion

1) Hard Assertion

b) Editable - toBeEditable()

b) Visible - toBeVisible ()

c) Enabled - toBeEnabled ()

d) Empty - toBeEmpty ()

e) URL - toHaveURL ()

f) Title - toHaveTitle ()

g) Text - toHaveText ()

h) Count - toHaveCount ()

i) Disabled - toBeDisabled ()

2.11 Watch Mode

- Project level
- Spec file level
- Test level

https://youtu.be/_tgNpYCIr8Y?si=sQMrNosIsUWWK6ox

2.12 Playwright UI

https://youtu.be/XVeUcQFVoMQ?si=DMiUHrU1BR8aBS8E

2.13 Trace Viewer - Test Traces, Actions, Metadata, Console, Log, Network etc.

https://youtu.be/isnkATBZ3BU?si=f-Cp8dDKJMM45Lve

Chapter- 03

3.1 Annotations

- Skip Test,
- Only (Run only selected test)

https://youtu.be/7p5xORtF_GY?si=MVxN6ZFRV6JXQaPj

3.2 How to Group & Run Tests in Playwright?

https://youtu.be/0jEWcCYPqS0?si=YHDLf-98ujs9sGhi

3.3 Tags in Playwright Test Automation?

https://youtu.be/qSkzQV3lXtM?si=KZt3uea3na1GZEvi

3.4 How to Run Same Test multiple times in Playwright?

https://youtu.be/W7B9sd7NtMM?si=rburClMjA5-QxiFp

3.5 How to Automatically Retry Test Execution When Test is Failed in Playwright?

https://youtu.be/LTujQSv8nAA?si=SU9JUuoOJuMtk7ol

3.6 How to Parametrize Tests in Playwright?

https://youtu.be/x_xmdPdH6Is?si=yYNfGzZ1l0RIJRkZ

3.7 How to Perform Visual Testing in Playwright?

- Testing Page Screenshot
- Testing Element Screenshot

https://youtu.be/WLWuzXiyxco?si=3UkmtnL6WNGXrNdK

3.8 Timeouts in Playwright?

- Test Timeout
- Assertion Timeout
- Action Timeout
- All Test Execution Timeout

https://youtu.be/kEwkksL3ZIA?si=9Rp17ZYfou2rJEXi

Chapter- 04

3.1 tsconfig.json file for Playwright with TypeScript

https://youtu.be/SUTPHOuMDlA?si=dBQQk6-T2CiUH0KG

The tsconfig.json file is the central place to configure TypeScript options for your project. Without it, TypeScript will default to certain behavior's, which might not always suit your needs, especially as your project grows.

Benefits:

* Improves IDE Experience - Type Checking: Autocomplete, Error Reporting etc.

* Project-wide Configuration

* Integration with Build Tools - webpack etc.

* Build and Compilation - TypeScript to JavaScript(using tsc)

Here's why you might want a tsconfig.json file:

- Specifies compiler options: You can define settings like target ECMAScript version ("target": "ESNext"), module system ("module": "CommonJS"), and other TypeScript-specific settings.
- Strict mode: You can enable strict mode and other TypeScript checks like strictNullChecks, noImplicitAny, forceConsistentCasingInFileNames, etc., to ensure type safety.
- Including/Excluding files: You can specify which files or directories to include or exclude from compilation, helping TypeScript avoid unnecessary processing of unwanted files.

tsconfig.json file

```
{
  "compilerOptions": {
    "target": "ESNext",            // Compile to the latest ECMAScript version
    "module": "CommonJS",            // Module system for Node.js compatibility
    "moduleResolution": "node",      // Resolve modules like Node.js
    "strict": true,                 // Enable all strict type-checking options
    "esModuleInterop": true,         // Allows default imports for CommonJS
modules
    "skipLibCheck": true,            // Skip type checking of declaration files for
faster builds
    "forceConsistentCasingInFileNames": true, // Ensure consistent file naming
    "outDir": "./dist",             // Output directory for compiled JavaScript files
    "rootDir": ".",             // Root directory for TypeScript files
    "resolveJsonModule": true,       // Allow importing of JSON files
    "types": ["node", "playwright"], // Include the types for Node.js and
Playwright
    "declaration": true,            // Generate .d.ts files for type declarations
    "sourceMap": true               // Generate source maps for easier debugging
  },
  "include": [
    "src/**/*.ts",   // Include all TypeScript files in the 'src' folder
    "tests/**/*.ts"  // Include all TypeScript files in the 'tests' folder
  ],
  "exclude": [
    "node_modules",  // Exclude node_modules from compilation
    "dist"          // Exclude the output directory
  ]
}
```

4.2 Page, Browser Context etc.

- How to create multiple browser sessions
- How to create multiple tabs/pages

https://youtu.be/WRA6T3MZJtc?si=ERvr29D9Rf_819ue

4.3 Rerun only failed Tests in Playwright?
https://youtu.be/Cbye79CPeFI?si=Pq4mDkj-7GM1UEyd

4.4 How to Handle Popups/Alerts/Dialogues in Playwright?

- Alerts
- Popups
- Prompt Popups

https://youtu.be/_q25VaWFMzU?si=iHMm6eNp14OM8kfe

4.5 How to Generate Multiple Types of Playwright Test Reports?

- HTML Test Report
- JSON Test report
- JUnit Test Report
- List Test Report
- Dot Test Report
- Allure Test Report

https://youtu.be/wKKmvf3w31E?si=oGiJZfR4XQQEq87f

4.6 How to Enable Video Recording in Playwright?
https://youtu.be/HrhgQtksxp0?si=pAJeSx1V2HOqOCIR

4.7 How to Perform Parallel/Parallelism Test Execution in Playwright?

- Config
- Command

https://youtu.be/LYX7AFEADfU?si=enEzQBV5gk3QqbUS

4.8 How to Generate Allure Report in Playwright?
https://youtu.be/348WHEvihFU?si=phpC7pnzxsS6jdCg

4.9 textContent (), innerText (), getAttribute()
https://youtu.be/PQ5UI0ElHcM?si=xjfo4z46mO14OMH0

4.10 How to Iterate through all the matching elements in Playwright.

- For of loop
- For loop
- For loop + nth() method

https://youtu.be/0bKIVa1qBfU?si=C2f1mgbavG3OZ3Fb

4.11 Handling Checkbox and Radio Buttons.

- Validate checked or unchecked
- How to check checkbox or radio buttons

https://youtu.be/go7iUlUUftM?si=geV8s6YTb1JlRo8s

Chapter- 05

5.1 How to Read .env File in Playwright with TypeScript?

- Plugin:

```
npm install dotenv –save
```

https://youtu.be/m4c_OZAXIz4?si=DP-FNSfmA_F3lf3n

5.2 How to Perform Data Driven Testing in Playwright with TypeScript Using JSON File?

- Create JSON file
- Read JSON file
- Pass each set of test data to playwright test

https://youtu.be/b05fB2M2Geo?si=7SwydxBRJZblwFan

- If any file import issue add below code in global.d.ts file & include file tsconfig.json file:

```
declare module '*.json' {
  const value: any;
  export default value;
}
```

5.3 How to Perform Data Driven Testing in Playwright with TypeScript Using CSV File?

- Plugin:

```
npm install csv-parse
```

- Create CSV file
- Create new spec file
- Read CSV file
- Pass each set of test data to playwright test

https://youtu.be/u7LPH_9xBrY?si=kbUkURjiemmoSyEt

- If any file import issue, add below code in global.d.ts file & include file tsconfig.json file:

```
declare module '*.csv' {
  const value: any;
  export default value;
}
```

5.4 How to Perform Data Driven Testing in Playwright with TypeScript Using EXCEL File?

- Plugin:

```
npm install xlsx
```

- Create Excel file with test data.
- Create utility function to read excel file.
- Create new spec file.
- Pass each set of test data to playwright test.

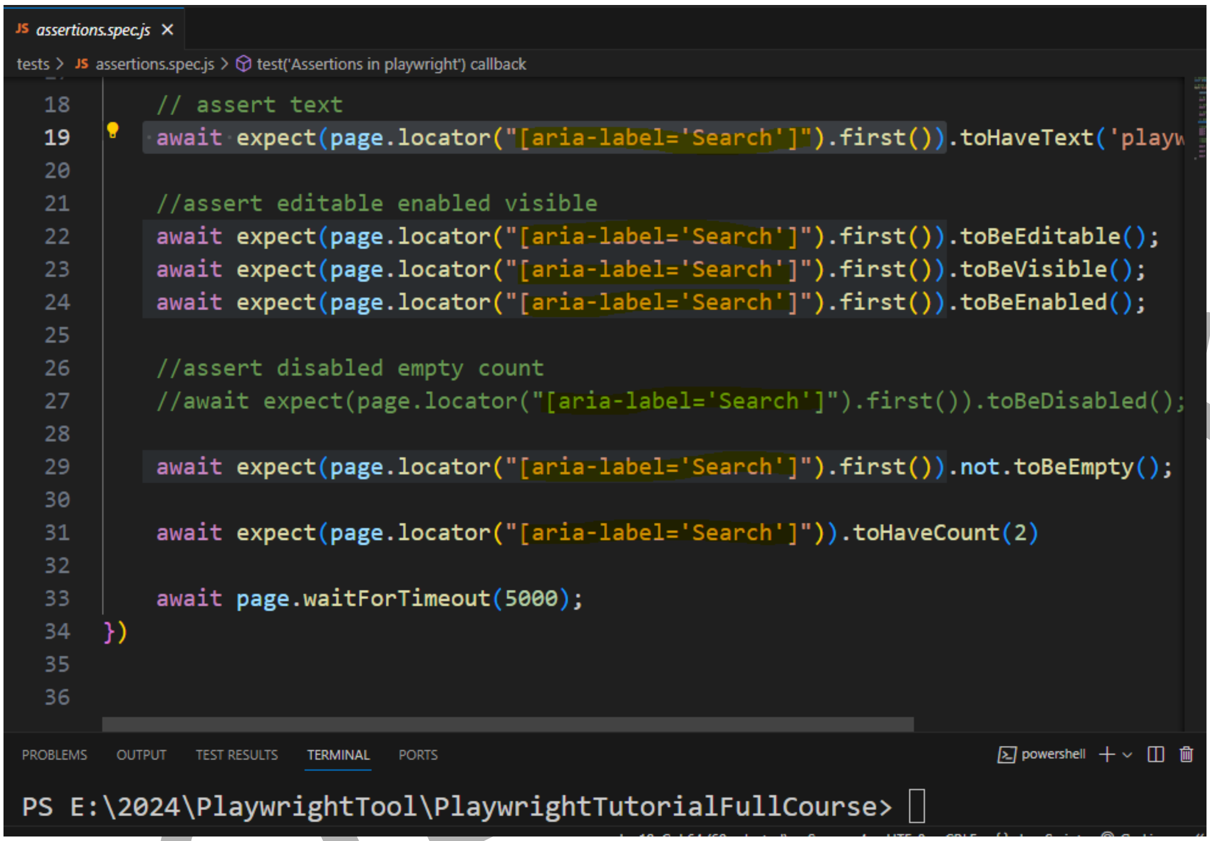
https://youtu.be/uTKfr_ipxFU?si=7zpkr7WdNmXp328p

5.5 Implement Page Object Model in Playwright

- Why Page Object Model?
- What is Page Object Model?
- Advantages & Disadvantages
- Sample Page Object Model Page Class
- Sample Page Object Model Playwright Test
- Use Case
- Create Pages
- Create Playwright Test using Page Object Model Design Pattern

https://youtu.be/TNuVQ-ArPsQ?si=-8Fsm63FyeyR9CmH

- Why POM (Page Object Model)?



assertions.spec.js

tests > assertions.spec.js > test('Assertions in playwright') callback

```
    // assert text
    await expect(page.locator("[aria-label='Search']").first()).toHaveText('playw

    //assert editable enabled visible
    await expect(page.locator("[aria-label='Search']").first()).toBeEditable();
    await expect(page.locator("[aria-label='Search']").first()).toBeVisible();
    await expect(page.locator("[aria-label='Search']").first()).toBeEnabled();

    //assert disabled empty count
    //await expect(page.locator("[aria-label='Search']").first()).toBeDisabled();

    await expect(page.locator("[aria-label='Search']").first()).not.toBeEmpty();

    await expect(page.locator("[aria-label='Search']")).toHaveCount(2)

    await page.waitForTimeout(5000);
})
```

```
PS E:\2024\PlaywrightTool\PlaywrightTutorialFullCourse>
```

- What is Page Object Model?

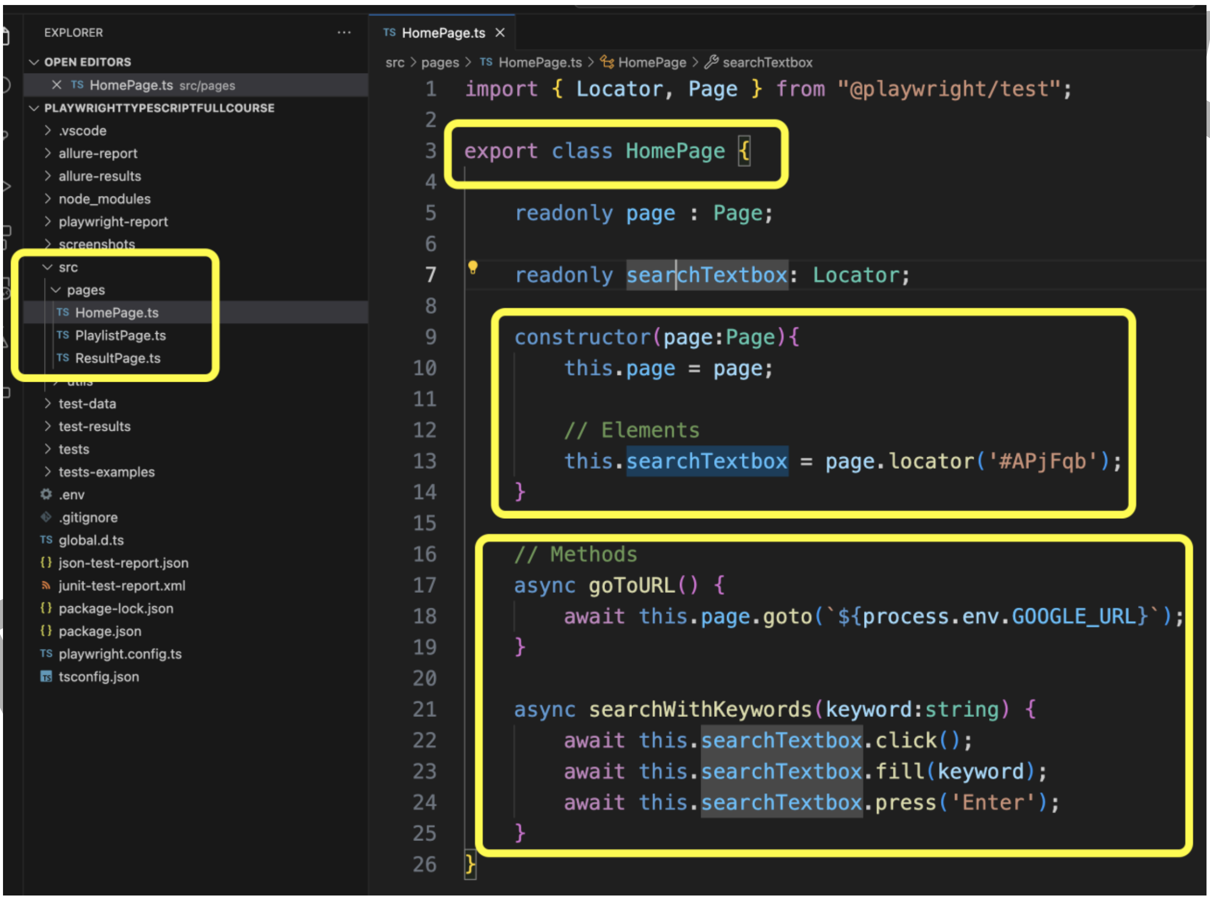
- Page Object Model is a design pattern to create object repository for web page elements
- For each web page there will be corresponding page class
- Page class contains all the web page elements of that page & also it contains page methods which perform operations on those web elements

- Advantages:

- Code will be cleaner & Easier to understand
- Object repository is independent of test scripts
- Test Script will be optimized because of elements respective abstraction methods in page classes

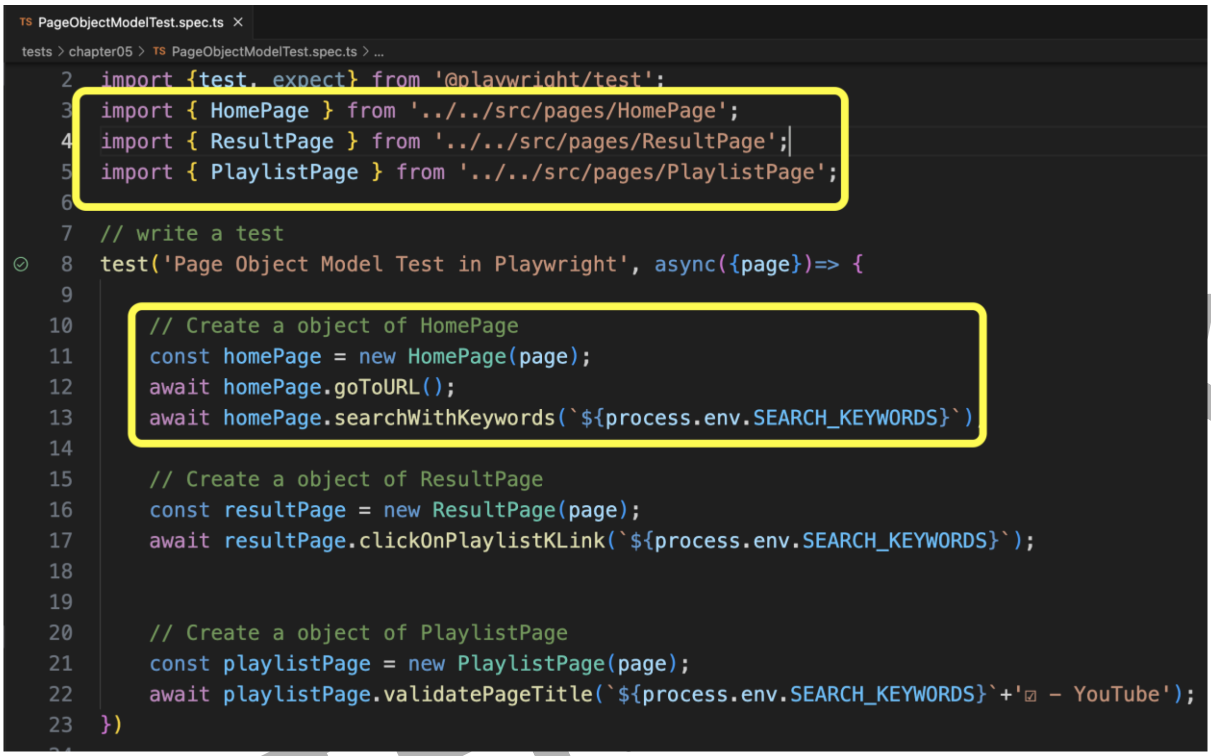
- Disadvantages
  - Time & Effort
  - Specific to project

- Sample Page Object Model Page Class:



```typescript
import { Locator, Page } from "@playwright/test";

export class HomePage {

    readonly page : Page;

    readonly searchTextbox: Locator;

    constructor(page:Page){
        this.page = page;

        // Elements
        this.searchTextbox = page.locator('#APjFqb');
    }

    // Methods
    async goToURL() {
        await this.page.goto(`${process.env.GOOGLE_URL}`);
    }

    async searchWithKeywords(keyword:string) {
        await this.searchTextbox.click();
        await this.searchTextbox.fill(keyword);
        await this.searchTextbox.press('Enter');
    }
}
```

- Sample Page Object Model Test:


PageObjectModelTest.spec.ts

tests > chapter05 > PageObjectModelTest.spec.ts > ...

```
import {test, expect} from '@playwright/test';
import { HomePage } from '../../src/pages/HomePage';
import { ResultPage } from '../../src/pages/ResultPage';
import { PlaylistPage } from '../../src/pages/PlaylistPage';

// write a test
test('Page Object Model Test in Playwright', async({page})=> {

    // Create a object of HomePage
    const homePage = new HomePage(page);
    await homePage.goToURL();
    await homePage.searchWithKeywords(`${process.env.SEARCH_KEYWORDS}`)

    // Create a object of ResultPage
    const resultPage = new ResultPage(page);
    await resultPage.clickOnPlaylistKLink(`${process.env.SEARCH_KEYWORDS}`);


    // Create a object of PlaylistPage
    const playlistPage = new PlaylistPage(page);
    await playlistPage.validatePageTitle(`${process.env.SEARCH_KEYWORDS}`+'☑ - YouTube');
})
```

5.6 How to Execute Only Changed Spec/Test Files in Playwright?

- Command:

```
npx playwright test - -only-changed
```

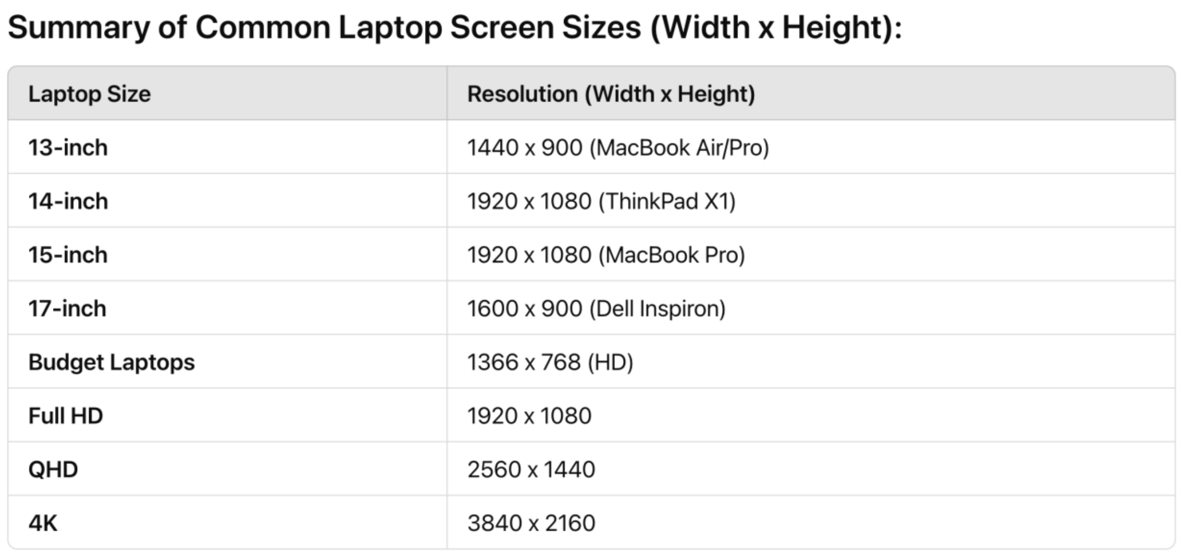
https://youtu.be/YmjQNbSWhcI?si=MhyAjlFcqM_e-6xK

5.7 How to maximize browser & set viewport size in Playwright?

- launchOptions

```
launchOptions : {
        args: ['--start-maximized']
    },
```

How to Set Viewport Size?



Summary of Common Laptop Screen Sizes (Width x Height):

Laptop Size	Resolution (Width x Height)
13-inch	1440 x 900 (MacBook Air/Pro)
14-inch	1920 x 1080 (ThinkPad X1)
15-inch	1920 x 1080 (MacBook Pro)
17-inch	1600 x 900 (Dell Inspiron)
Budget Laptops	1366 x 768 (HD)
Full HD	1920 x 1080
QHD	2560 x 1440
4K	3840 x 2160

https://youtu.be/xTeFNFtoofk?si=BFj_dC4-m1turWvF

5.8 Fixture -

- How to implement Global Hooks - BeforeEach & AfterEach

https://youtu.be/HqJ0LRy5KME?si=rLmpcWo91jjb1i4u

5.9 How to Optimize Page Object Model Tests with Fixture

https://youtu.be/jftW1abJqJE?si=LEIzxjZPheBmeTTj

5.10 Run Tests on Multiple Environments – QA, STAGE, DEV, PROD etc.

- Create environment variable
- Create test data for each environment
- Create an interface for type safety
- Create utility method to read all the files
- Create playwright test
- Run playwright test

https://youtu.be/w3pR73rsKBU?si=ykKOuz8_QLTyVgLH

5.11 Global Setup & TearDown

- How to Save Authentication?

Chapter- 06

6.1 Create POST API Request using Static Request Body in Playwright & TypeScript

- Use Case
- Create JSON file
- Create Spec file
- Automate POST API request
    - Print JSON API response
    - Validate status code, status text, response type.
    - Validate property/keys
    - Validate API response body

https://youtu.be/k8IHwDrqVxU?si=HhNRd60eIx3Hcf3Z

6.2 Create POST API Request using Dynamic Request body in Playwright & TypeScript

- Prepare dynamic API request body JSON file
- Create utility method
- Automate POST API request
    - Print JSON API response
    - Validate status code, status text, response type.
    - Validate property/keys
    - Validate API response

https://youtu.be/MXH2MNPHkFw?si=iwVc-CTUncbiiAtz

6.3 How to Create Dynamic POST API Request using Playwright & TypeScript

- Plugin:

  npm install @faker-js/faker

https://youtu.be/zXCARAWdmv4?si=jmd3276RHzbFQmoQ

6.4 Create Dynamic POST API Request with TypeSafety using Playwright & TypeScript

https://youtu.be/k4V5MdiaqmE?si=NQT1ceeow8-KB0gb

6.5 How to Create GET API Request using Playwright & TypeScript?

- Manual Use Case
- Automate GET API Request

https://youtu.be/v6kkPVf1bbU?si=hZG_OKGAuZSHOwaz

6.6 How to Pass Query Parameters in Playwright API Automation Testing?

- Manual Use Case
- Automate GET API Request using Query Parameters

https://youtu.be/cw-ZbZjhJFM?si=_ewDXxbFpDiSIiMh

6.7 How to Create PUT API Request using Playwright & TypeScript

- Manual Use Case
- Generate token
- Automate PUT API Request

https://youtu.be/nLFZzpLhw2o?si=MNbLjtj6xr6e6sm0

6.8 How to Create PATCH API Request using Playwright & TypeScript

- Manual Use Case

- Automate PATCH API Request

https://youtu.be/6gE-mf68duA?si=eNDCXzi3NLYTRIuN

6.9 How to Create DELETE API Request using Playwright & TypeScript

- Manual Use Case

- Automate End to End API Testing flow

https://youtu.be/VcidovRv_Ac?si=R5xP8QZlxpi5Axts

6.10 What is API Mocking?

https://youtu.be/ufKlOqH5AJE?si=NKpQ5qpLEFOXzLPb

API mocking is a technique to simulate the behavior of an API or service without calling actually API. This is extremely beneficial in various ways such as Isolation, Speed & Control etc.

Example: Banking Transactions, Gift Card & Third-Party API's etc.

- API Mocking vs API Testing?

- 3 Ways of API Mocking in Playwright

- API Mocking vs API Testing

Aspect	API Mocking	API Testing
Purpose	Simulate the behavior of an API without actual backend logic.	Verify the actual behavior, performance, and functionality of an API.
When to Use	When the real API is unavailable or incomplete.	When the real API is ready and needs validation.
Environment	Isolated, often in development or early stages.	QA or Staging environment where the actual API is deployed.
Focus	Creating controlled, predictable responses.	Ensuring that the API works correctly in various scenarios.
Response Behavior	Predefined, simulated responses based on inputs.	Real responses from the actual API, depending on the backend logic.
Network Calls	No real network calls; local mock server or tool.	Real network calls to the actual API endpoint.
Error Handling	Simulated errors or predefined failure responses.	Testing real error scenarios (e.g., 500 errors, network issues).
Testing Types	Typically used in unit/integration development phases.	Covers functional, performance, security, and load testing.
Tools	WireMock, Mockoon, Mountebank, Postman (mock mode).	Postman, SoapUI, RestAssured, JUnit, Jest, etc.

Use Cases	Early-stage development, testing interactions before API completion.	Verifying full functionality, security, and performance of a live API.
Data Source	Static or predefined data responses.	Real-time data from the API (e.g., database queries, live data).
Cost	Low cost as no real API calls are made.	May incur costs if testing against paid or rate-limited APIs.
Flexibility	High flexibility in simulating different scenarios and responses.	Less flexibility; testing is dependent on the actual API and its limitations.

- 3 Ways of API Mocking in Playwright

- API Requests
- API Response
- API Mocking from HAR Files
    - Recording, Modifying & Replaying

6.11 How to Mock API Requests in Playwright?

https://youtu.be/zX1AiNF3USM?si=bfzNJFGczh2_HPFD

6.12 How to Mock API Response in Playwright?

https://youtu.be/16gbSnCW71c?si=8hmlH9iTkqeMe2se

6.13 How to Record .HAR file, Modify .HAR file & Replaying from .HAR file in Playwright & TypeScript

https://youtu.be/1SyOKq3bH4U?si=UFXQSM1X96wItD2k

Chapter- 07

7. Jenkins, Azure DevOps Pipeline & GitHub Actions

7.1 How to Integrate Playwright with Jenkins + GitHub

- How to Create GitHub Repository & Push Code?

https://youtu.be/KTlL0-64K9I?si=yxGhfdLCnVMkK2GF

- Generate PAT (Personal Access Token) for GitHub?

https://youtu.be/-kut2qhCHlA?si=khQ9xa9zguyNJADo

- How to Download & Setup Jenkins?

https://youtu.be/qzNerpJQevM?si=48lrvKsM43JLA4wj

- Install Plugins:

- NodeJS Plugin, Git Plugin, JUnit Plugin & Allure Plugin.

https://youtu.be/v4SW5pSIHL0?si=e_k2yvqhMdyBt3Dp

- How to Add GitHub PAT token in Jenkins?

https://youtu.be/_5k9irhkkOg?si=phm85rgHqEvVdRdB

- How to Create Jenkins Job & Run Playwright Tests?

https://youtu.be/zALfXCHv1Ko?si=XpyUJ0CQXInUaLZ4

- How to Attach Artifacts (playwright-report)

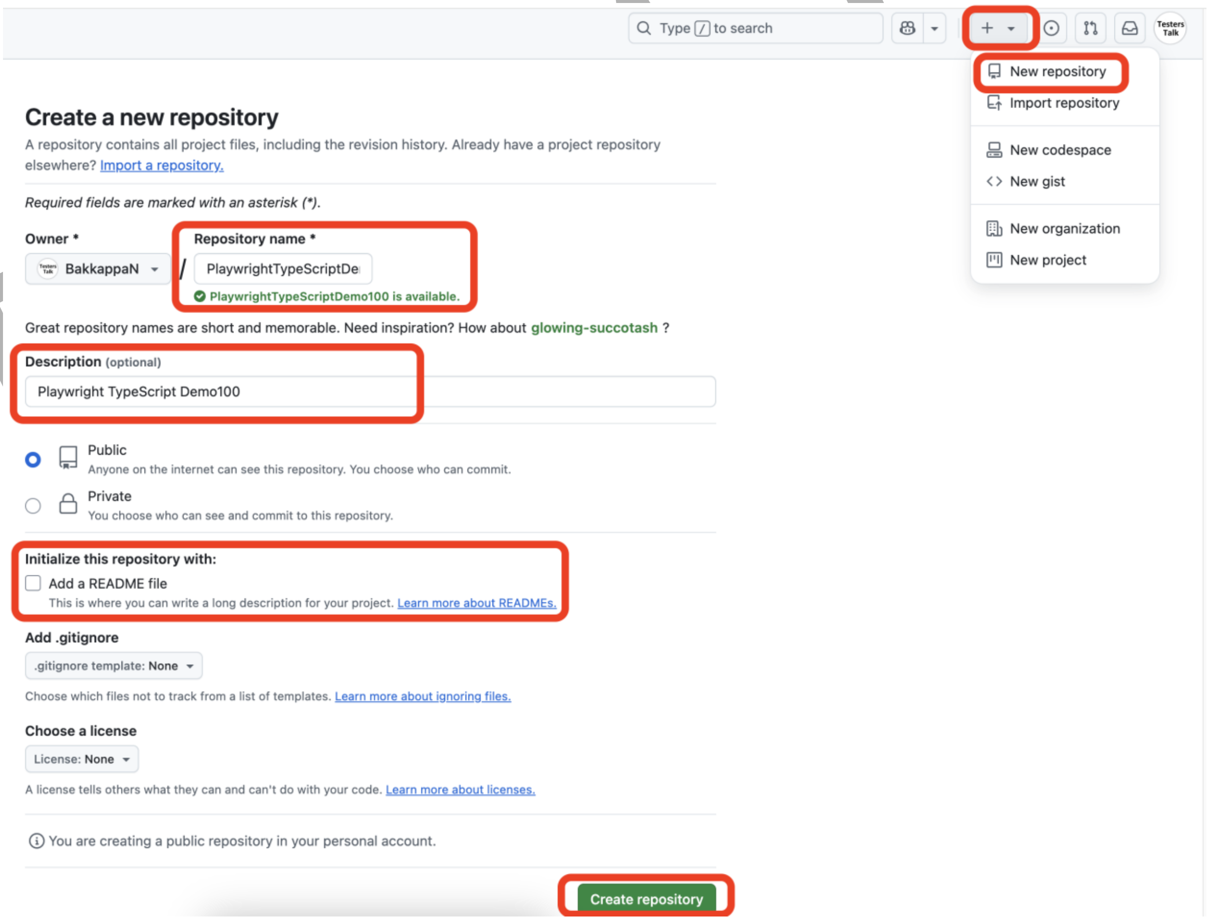
https://youtu.be/GvdEDXcY7O8?si=g1cjI1cvCbE1MulE

- How to Publish Junit Test Results in Jenkins?

https://youtu.be/dujbT9s16gM?si=XLHfk6ouKba4qY3j

- How to Publish Allure Report in Jenkins?

https://youtu.be/SDzIsiPV6Cw?si=nkaUIWbLgDJPC-_y



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner *: BakkappaN / Repository name *: PlaywrightTypeScriptDe

PlaywrightTypeScriptDemo100 is available.

Great repository names are short and memorable. Need inspiration? How about glowing-succotash ?

Description (optional): Playwright TypeScript Demo100

● Public — Anyone on the internet can see this repository. You choose who can commit.

○ Private — You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file — This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore: .gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license: License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

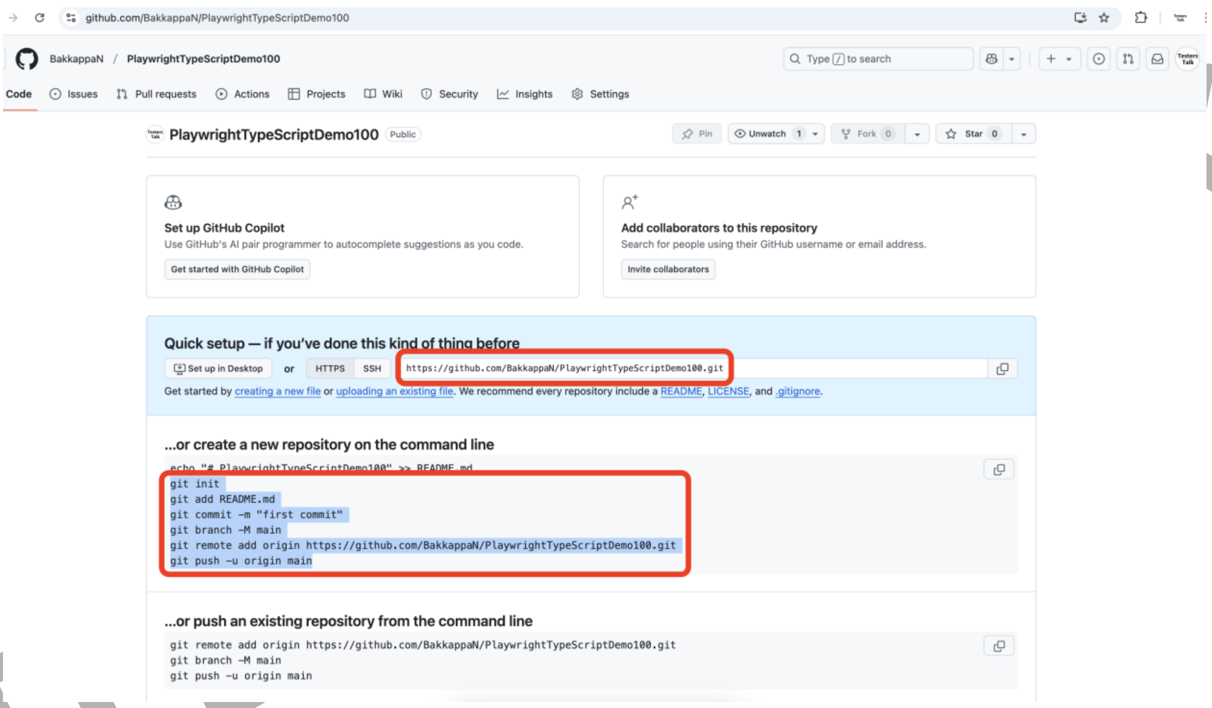
You are creating a public repository in your personal account.

Create repository

-> How to Create GitHub Repository?

-> How to push source code to GitHub Repository?

```
git init
git add *
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/BakkappaN/
git push -u origin main
```


github.com/BakkappaN/PlaywrightTypeScriptDemo100
BakkappaN / PlaywrightTypeScriptDemo100
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
PlaywrightTypeScriptDemo100
Public
Pin
Unwatch 1
Fork 0
Star 0

Set up GitHub Copilot

Use GitHub's AI pair programmer to autocomplete suggestions as you code.

Get started with GitHub Copilot

Add collaborators to this repository

Search for people using their GitHub username or email address.

Invite collaborators

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/BakkappaN/PlaywrightTypeScriptDemo100.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# PlaywrightTypeScriptDemo100" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/BakkappaN/PlaywrightTypeScriptDemo100.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/BakkappaN/PlaywrightTypeScriptDemo100.git
git branch -M main
git push -u origin main
```

-> Steps to Generate GitHub Personal Access Token (PAT)

1. Go to GitHub and log in to your account.
2. Click on your profile picture in the top-right corner and select Settings.
3. In the left sidebar, click Developer settings.
4. Click Personal access tokens.
5. Click Generate new token.
6. Provide a descriptive name for the token (e.g., "Jenkins CI Token").
7. Select the required scopes (e.g., repo, workflow, or other necessary permissions).
8. Click Generate token.
9. Copy the token immediately as you won't be able to view it again.

7.2 How to Integrate Playwright with Azure DevOps Pipeline

- Download & Install GIT - https://git-scm.com/downloads
- How to Install Azure Repos extension in VS Code
- How to Create New Repository in Azure DevOps?

https://youtu.be/SxLA43-JAqU?si=f_SWhZh8539hRsYj

- How to Push Source Code from VS Code to Azure DevOps?

https://youtu.be/5qWC0rNip9g?si=XYfyd8lvjyAZhnRL

- How to Create .yml(YAML) file?

https://youtu.be/ODGKrCKx0xY?si=2pssTMSqiLEatyT-

- How to Create Azure DevOps Pipeline?

https://youtu.be/hfcR3Kxr-lU?si=HLSplj1Tf2dsUb_u

- How to Run Playwright Tests using ADO pipeline?

https://youtu.be/y_KsVBSdZs8?si=11enVfed4v_82iXg

- How to Publish JUnit Test Results in ADO?

https://youtu.be/y_KsVBSdZs8?si=rmKaI-BBkNDhEcf7

- How to Attach "playwright-report" in ADO as Artifacts?

https://youtu.be/KiOJCTYaePc?si=5806zlmv4WPKXuL8

- How to Update Test Case Status Automatically in Test Plan (Pipeline + Local)? https://youtu.be/aT_1uulU2dE?si=l6lRTRTTFCzO8Z2y

7.3 Integrate Playwright with GitHub Actions:

- How to Create a .yml(YAML) File?

https://youtu.be/mAP7yRviP0k?si=SADoKF6VrVCIAwC0

- How to Run Playwright Tests using GitHub Actions?

- Workflow - Auto Trigger
- Workflow - Manual Trigger

- How to Attach “playwright-report“ as Artifacts

https://youtu.be/eU97jYluRPA?si=CgOn1jLkKA19ksnr

https://youtu.be/gI3L2AXndFE?si=o3dTlq_IiSmJfJbU

THE END…