**1. LearningPlan\_Playwright\_UI**

**Content:**

* **Overview**: Introduction to Playwright and its capabilities for UI automation.
* **Topics**:
  1. Basics of Playwright: Installation, setup, and first test.
  2. Handling dynamic elements and synchronization.
  3. Working with different browser contexts and incognito modes.
  4. Strategies for dealing with flaky tests.
  5. Implementing the Page Object Model (POM) design pattern.
  6. Capturing screenshots and videos for debugging.
* **Exercises**:
  1. Automate a login form for a sample website.
  2. Create a test that validates navigation between pages.
  3. Implement POM for a multi-step workflow.
* **Resources**:
  1. Official Playwright Documentation: [https://playwright.dev](https://playwright.dev/)
  2. Tutorials and examples from GitHub.

**2. LearningPlan\_Playwright\_API**

**Content:**

* **Overview**: Automating API testing with Playwright.
* **Topics**:
  1. Sending HTTP requests using Playwright.
  2. Validating API responses.
  3. Integrating API tests into UI workflows.
  4. Error handling for API failures.
  5. Mocking API responses for testing.
* **Exercises**:
  1. Write a test to validate a sample REST API.
  2. Mock an API response and verify UI behavior.
  3. Integrate an API call into a Playwright test for end-to-end validation.
* **Resources**:
  1. Playwright API Testing Guide: [Link](https://playwright.dev/docs/network)
  2. Postman for API Testing.

**3. LearningPlan\_Playwright\_Mobile**

**Content:**

* **Overview**: Responsive testing and mobile emulation in Playwright.
* **Topics**:
  1. Configuring device emulation for mobile testing.
  2. Handling viewport sizes and responsive designs.
  3. Testing Progressive Web Apps (PWAs).
  4. Debugging mobile-specific features.
* **Exercises**:
  1. Automate a test for a mobile login page.
  2. Validate responsiveness of a website across multiple devices.
  3. Test a PWA for offline capabilities.
* **Resources**:
  1. Playwright Mobile Emulation: [Link](https://playwright.dev/docs/emulation)
  2. Browser Developer Tools for mobile debugging.

**4. Playwright\_Framework\_Design**

**Content:**

* **Overview**: Building scalable test automation frameworks with Playwright.
* **Topics**:
  1. Structuring the test project.
  2. Setting up reusable utilities and helper functions.
  3. Configuring environment variables and test data management.
  4. Implementing parallel execution and sharded testing.
  5. Integrating test reporting and logging tools.
* **Exercises**:
  1. Design and implement a basic test framework for a sample application.
  2. Add a logging mechanism to capture test execution details.
  3. Configure parallel execution for faster test runs.
* **Resources**:
  1. Playwright Framework Examples on GitHub.
  2. Allure Reporting: [Link](https://docs.qameta.io/allure/)

**5. Playwright\_Advanced\_Features**

**Content:**

* **Overview**: Exploring advanced capabilities of Playwright.
* **Topics**:
  1. Cloud execution using platforms like SauceLabs or BrowserStack.
  2. Visual regression testing with Playwright.
  3. Integrating Slack notifications for test failures.
  4. Running tests in headless mode for CI/CD.
* **Exercises**:
  1. Set up a test suite to run on BrowserStack.
  2. Implement visual regression testing for a sample application.
  3. Configure Slack notifications for test execution results.
* **Resources**:
  1. Playwright Visual Comparisons: [Link](https://playwright.dev/docs/test-snapshots)
  2. BrowserStack Integration: [Link](https://www.browserstack.com/docs/automate/playwright)

**6. ScenarioBased\_InterviewQuestions\_Playwright**

**Content:**

* **Overview**: Common scenario-based interview questions for Playwright.
* **Questions & Answers**:
  1. **Dynamic Elements**: How do you handle elements that take time to appear?
     + **Answer**: Use waitForSelector or waitForLoadState methods.
  2. **Error Handling**: How do you handle flaky tests?
     + **Answer**: Use retries, proper waits, and debugging tools to identify root causes.
  3. **API Integration**: How do you validate API responses in UI tests?
     + **Answer**: Use page.request to make API calls and validate the responses.
  4. **Cross-Browser Testing**: How do you ensure compatibility across browsers?
     + **Answer**: Use Playwright’s browser context and test on Chromium, Firefox, and WebKit.
* **Tips**:
  1. Practice coding exercises for real-world scenarios.
  2. Focus on problem-solving and debugging techniques.
* **Resources**:
  1. Playwright Interview Questions on Medium and LinkedIn.

**7. CapstoneGuide\_Playwright**

**Content:**

* **Overview**: A comprehensive guide to building a capstone project using Playwright.
* **Project**: Automating a Travel Booking Application.
* **Steps**:
  1. Define test scenarios (e.g., booking a flight, adding a hotel, making a payment).
  2. Design a test framework using POM.
  3. Automate multi-step workflows with validations.
  4. Execute tests across multiple browsers and devices.
* **Deliverables**:
  1. Test cases and scripts for each feature.
  2. Test execution reports.
  3. Documentation for the framework.
* **Tips**:
  1. Start with small, modular tests and integrate them into a larger workflow.
  2. Use debugging and reporting tools for better insights.
* **Resources**:
  1. Sample Travel App for Testing: [Link](https://demo.guru99.com/test/newtours/)
  2. Playwright GitHub Examples.

**Additional Resources for Learning Playwright**

1. **Official Documentation**:
   * Playwright Docs: <https://playwright.dev/docs/>
   * API Reference: <https://playwright.dev/docs/api/class-playwright>
2. **Video Tutorials**:
   * YouTube Playlist: [Playwright Tutorials](https://www.youtube.com/results?search_query=playwright+tutorial)
   * FreeCodeCamp Playwright Guide: [Link](https://www.freecodecamp.org/news/playwright-tutorial/)
3. **GitHub Repositories**:
   * Sample Projects: <https://github.com/microsoft/playwright>
   * Playwright Examples: <https://github.com/playwright-community>
4. **Learning Platforms**:
   * Udemy: [Playwright Courses](https://www.udemy.com/topic/playwright/)
   * Pluralsight: [Playwright Testing](https://www.pluralsight.com/)
5. **Blogs and Articles**:
   * Medium: [Playwright Tutorials](https://medium.com/tag/playwright)
   * Dev.to: [Playwright Articles](https://dev.to/t/playwright)