3. Interview Prep: Questions & Answers

Theory Questions

- **What is E2E testing?**
  Validates complete workflows from the user's perspective, ensuring frontend, backend, and database work together.
- **Why Playwright for E2E?**
  Supports multiple browsers, handles async, allows network interception, integrates into CI/CD.
- **Difference: UI vs API vs E2E testing?**
  - UI: Focus on individual interface elements.
  - API: Validate backend endpoints and responses.
  - E2E: Full workflow validation across frontend & backend.
- **Positive vs Negative tests?**
  - Positive: Expected input/action → expected result.
  - Negative: Invalid input, errors, or edge cases → handled gracefully.
  -

Practical Questions

- **How do you handle async content?**
  `page.waitForSelector()`, `expect(locator).toBeVisible()`, `page.waitForResponse()`.
- **How to validate login success?**
  Check a unique success indicator, e.g., `.auth-badge`.
- **How to test form submission?**
  Use `page.fill()` → `page.click()` → validate success dialog or backend.
- **How to validate navigation?**
  Click links/buttons, assert URL/header, check page-specific elements.
- **How to handle flaky tests?**
  Add waits for async operations, retries, clean test data, and isolate tests.
- **CI/CD integration for Playwright?**
  Run `npx playwright test --reporter=html` in pipeline; fail build if tests fail.
  -

4. Developer Notes / Best Practices

- **Test Data Management**
  - Use isolated test accounts or test databases.
  - Reset or clean data between runs to avoid conflicts.
- **Async Handling**
  - Always wait for elements or API responses before assertions.
  - Avoid hardcoded `sleep()`; use Playwright's built-in waiting.
- **Selectors & Locators**
  - Use stable, unique selectors (`data-testid` recommended).
  - Avoid fragile CSS/XPath that changes often.
- **Repeatable & Idempotent Tests**
  - Tests should pass on repeated runs without manual intervention.
  - Clean up created bookings or reset form state.
- **Error Handling & Logging**
  - Log errors and test steps for debugging.
  - Capture screenshots or videos on failure.
- **Performance Considerations**
  - Avoid unnecessary waits or long test sequences.
  - Use parallel workers for faster execution.
- **CI/CD Best Practices**
  - Run E2E tests on pull requests to catch regressions.
  - Generate HTML reports for easy review.
- **Maintenance & Versioning**
  - Keep selectors, flows, and test data up-to-date.
  - Regularly review for deprecated API usage or broken flows.
  - .

#	Test Name / Scenario	Input	Expected Result	Actual Status	Type of Test	Positive / Negative	Notes / Theory
1	Login – should login successfully with valid	Enter valid username & password	User authenticated, success indicator appears	❌ Not found: text=Authenticated ✅ Found	UI / Functional	Positive	Verifies login works; success indicator can be UI

	credentials			`.auth-badge`			element or text.
2	Login – should show error message with invalid credentials	Enter invalid username/password	Error message displayed	✅ Error message found `.message.error`	UI / Validation	Negative	Ensures proper feedback for invalid login attempts.
3	Booking Form – should fill and submit booking form	Fill all required fields, click submit	Booking created, confirmation dialog shown	✅ Booking form submitted successfully	UI / Functional	Positive	Verifies booking form submission and success feedback.
4	Booking Details – should display complete booking details	Open booking details page	Booking information visible	✅ Booking details displayed successfully	UI / Functional	Positive	Ensures all booking info is correctly rendered.
5	Booking Details – should show edit button when authenticated	User logged in, open details	Edit button visible	✅ Edit button visible	UI / Functional	Positive	Only authenticated users can see/edit bookings.
6	Booking Details – should navigate to edit view	Click edit button	Navigates to edit page	✅ Successfully navigated to edit view	UI / Navigation	Positive	Validates navigation from details → edit view.
7	Booking Details – should refresh	Click refresh button	Booking details updated	✅ Refresh button works	UI / Functional	Positive	Ensures refresh reflects latest

	booking details						booking info.
8	Booking Details – should navigate back from details to list	Click back button	Returns to bookings list	✅ Back navigation works - heading found	UI / Navigation	Positive	Validates back navigation is functional.
9	Bookings List – should display booking list	Open bookings page	List/table rendered	⚠️ May need wait for async load	UI / Functional	Positive	Ensures booking list displays; async waits may be needed.
10	Bookings List – should create new booking	Click create, fill form, submit	Booking added successfully	✅ Booking creation attempted	UI / Functional	Positive	Confirms new booking creation functionality.
11	Bookings List – repeated booking creation	Click create, fill form	Booking added	✅ Booking creation attempted	UI / Functional	Positive	Validates repeated creation works reliably.
12	Bookings List – refresh / load booking list	Reload page / click refresh	Updated list displayed	✅ Booking list refreshed	UI / Functional	Positive	Ensures booking list updates properly.
13	Navigation – general page navigation step 1	Click links/buttons	Navigate to correct page	✅ Navigation works	UI / Navigation	Positive	Validates navigation links/buttons.
14	Navigation – general	Click links/buttons	Navigate to	✅ Navigati	UI / Navigation	Positive	Confirms navigati

	page navigation step 2		correct page	on works			on flow consistency.
15	Navigation – general page navigation step 3	Click links/buttons	Navigate to correct page	✅ Navigation works	UI / Navigation	Positive	Ensures all main navigation paths function.
16	Booking Form – repeated submission	Fill form again, submit	Booking created, confirmation shown	✅ Booking form submitted successfully	UI / Functional	Positive	Verifies repeated form submissions work without errors.

Start E2E Tests

[Login Page]

- Enter credentials

- Validate success/error feedback

[Booking Form Page]

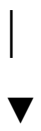
- Fill required fields

- Submit form

- Validate success dialog

[Booking Details Page]

- Open details for a booking
- Check information displayed
- Show edit button (auth required)
- Navigate to edit view
- Refresh booking details
- Navigate back to list

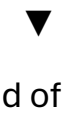


[Bookings List Page]

- Open booking list
- Display all bookings
- Create new booking(s)
- Refresh booking list

[Navigation Validation]

- Navigate between pages
- Validate links/buttons
- Ensure all flows work



End of E2E Tests

- All assertions complete