AI-Accelerated Development Documentation

Objective

The objective of this document is to demonstrate how **AI tools were used responsibly and effectively** to accelerate the development of a Playwright-based UI automation framework, while ensuring code correctness, security, and best practices.AI was used as a **development assistant**, not as a replacement for engineering judgment or testing validation.

AI Tools Used

- **ChatGPT** – for help with automation design, selectors, debugging, and refactoring
- **AI-assisted IDE suggestions** – for quicker code writing and cleanup

I didn't share any sensitive data, credentials, or private information with AI tools.

Areas Where AI Accelerated Development

Project Scaffolding and Architecture

AI was used to quickly scaffold a clean and maintainable project structure, including:

- Separation of concerns using the **Page Object Model**
- Utility layers for environment variables, test data, and money parsing
- Test folder organization aligned with Playwright best practices

This reduced setup time significantly and allowed focus on business logic and test validation.

Locator Strategy and Selector Stability

AI assisted in recommending **Playwright best practices**, such as:

- Using `getByRole()` and `getByLabel()` for user-centric, accessible selectors

- Avoiding brittle selectors like `nth-child()` or deeply nested CSS
- Simplifying selectors by removing unnecessary regular expressions where not needed

All suggested selectors were manually verified against the actual DOM.

Test Logic and Price Calculation Validation

AI helped outline the logical approach for validating:

- Line item total = unit price × quantity
- Cart subtotal = sum of all line totals

Based on this guidance, custom utility methods were implemented to:

- Parse currency values from the UI
- Handle rounding consistently
- Assert correctness of calculated totals

All calculations were verified by executing tests locally.

Validation and Quality Assurance

To ensure AI-assisted output met professional standards:

- All tests were executed locally to validate behavior
- Assertions were added for critical checkpoints (login, cart totals, order success)
- Code was refactored to improve readability and reduce duplication
- Silent error handling was avoided where failures should surface
- Selector reliability was reviewed and improved manually

What AI Did NOT Do

- AI did not replace manual testing or validation
- AI-generated code was not used blindly; all outputs were reviewed and adapted
- Final responsibility for correctness and quality remained with the developer

Conclusion

AI tools significantly accelerated development by reducing repetitive work and providing guidance on structure, selectors, and reporting. Final implementation quality was ensured through execution, validation, refactoring, and adherence to security and automation best practices.

AI acted as a **productivity enhancer**, while engineering judgment and ownership remained central to the development process.