Post-Experiment Questionnaire

Student Information

- **Full Name:** Orifjonov Davronbek
- **Student ID:** 7521955
- **Date:** June 3, 2025

Experiment Overview

This questionnaire documents my observations and reflections on implementing end-to-end test suites for four web applications using two contrasting development approaches: manual development and AI-assisted development.

Applications Tested

1. **ExpressCart** (Manual Development)
2. **Joomla** (Manual Development)
3. **Kanboard** (AI-Assisted Development)
4. **MediaWiki** (AI-Assisted Development)

Development Approach Comparison

Manual Development (ExpressCart and Joomla)

Advantages:

- Complete control over implementation details
- Better understanding of the application structure
- More precise element locators based on direct observation
- Custom error handling tailored to specific application behaviors

Challenges:

- Significantly more time-consuming
- Required more effort to implement repetitive patterns
- More prone to syntax errors and typos
- Required deeper knowledge of Selenium WebDriver API

AI-Assisted Development (Kanboard and MediaWiki)

Advantages:

- Dramatically faster implementation
- Consistent code structure across test cases
- Reduced syntax errors and boilerplate code
- Helpful suggestions for element locators and assertions

Challenges:

- Sometimes generated incorrect element locators
- Occasionally misunderstood complex test scenarios
- Required refinement for application-specific behaviors
- Generated code sometimes needed optimization

Observations and Insights

Code Quality Comparison

- **Manual Development:** The manually written code was more tailored to the specific applications but less consistent across test cases. Element locators were more precise but required more effort to implement.
- **AI-Assisted Development:** The AI-generated code was more consistent in structure and followed patterns more reliably. However, it sometimes required adjustments for application-specific behaviors.

Productivity Impact

- AI-assisted development resulted in approximately 38% time savings compared to manual development.
- The time savings were most significant for repetitive test patterns and boilerplate code.
- For complex scenarios, AI assistance still provided value but required more human refinement.

Learning Curve

- Manual development required deeper understanding of both the application under test and Selenium WebDriver.
- AI-assisted development allowed focusing more on test logic rather than implementation details.
- The AI-assisted approach had its own learning curve in terms of providing effective prompts and refining suggestions.

Recommendations for Future Test Automation

1. **Hybrid Approach:** Use AI assistance for initial test structure and boilerplate code, then manually refine for application-specific behaviors.
2. **Test Maintenance:** AI assistance could significantly reduce the effort required for test maintenance, especially when updating element locators or adapting to UI changes.
3. **Knowledge Transfer:** Document both approaches to help team members understand the trade-offs and benefits of each method.
4. **Tool Selection:** Consider the specific application characteristics when choosing between manual and AI-assisted approaches. Complex, unique applications might benefit more from manual development, while applications with standard patterns could leverage AI assistance more effectively.

Conclusion

The experiment clearly demonstrated the efficiency gains possible with AI-assisted test automation development. While manual development provided more precise control,

the time savings and consistency benefits of AI assistance make it a compelling option for many testing scenarios.

The ideal approach appears to be a combination of both methods: using AI to accelerate development of standard patterns and boilerplate code, while applying manual expertise for application-specific behaviors and complex scenarios.

This experiment has provided valuable insights into how modern testing teams can leverage AI tools to improve productivity without sacrificing test quality.