Developing Automated UI Testing for Ara ICT Division

Daniel Wild

Ara Institute of Canterbury

Te Pūkenga

d.j.wild2996@gmail.com

Dr. Luofeng Xu

(Supervisor)

Ara Institute of Canterbury

Te Pūkenga

Luofeng.Xu@ara.ac.nz

Vinay Varma

(Supervisor)

Ara Institute of Canterbury

Te Pūkenga

*Vinay.Varma@ara.ac.nz*

ABSTRACT

This paper describes the project done in partnership with the Ara ICT Division. Ara Institute of Canterbury Te Pūkenga has a series of react-based web apps for which the ICT Division are attempting to develop a set of automated deployment UI tests that require minimal interference. To ensure that no release degrades the overall standard of the tested apps. This was achieved using Puppeteer, a Node.js library that provides a high-level API to control Chrome development tool protocols. As a result, each app had an automated testing suite integrated into its deployment pipeline, ensuring that software behaves as it should and that new bugs have not been introduced after an update.

**Keywords**: Automation, Testing, Puppeteer, Agile, Kanban, CI/CD Pipeline

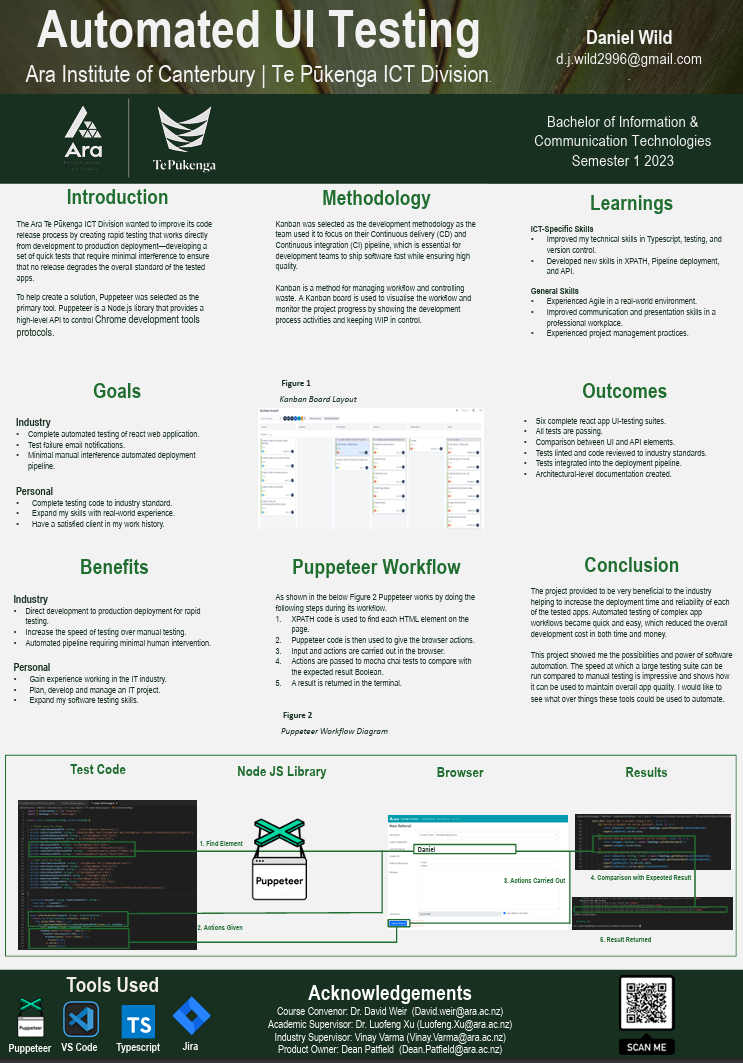
This quality-assured poster paper appeared at the 13th annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ 2022) and the 35th Annual Conference of the National Advisory Committee on Computing Qualifications [hosted in Christchurch], October 4 - 7.

# INTRODUCTION

Ara Institute of Canterbury is a subsidiary of Te Pūkenga New Zealand Institute of Skills and Technology (2022) in Canterbury, New Zealand. Ara provides a personalised learning experience. One that is flexible and respectful of students’ educational needs and career goals. The focus is always on balancing theory with practical learning to ensure students are work-ready when they graduate.

**Figure 1**

*Automated UI Testing Short Paper Poster*



# Project Details

The motivation for the project comes from the industry’s need to find a way to improve its code release process by creating rapid testing that works directly from development to production deployment—developing a set of quick tests that require minimal interference.

Project Scope

Creating six automated UI testing suites ensures that software behaves as it should and that new bugs have not been introduced after an update to integrate each into the six apps deployment pipeline to ensure that no release degrades the overall standard of the apps.

Industry Goals

* Complete automated testing of react web application.
* Test failure email notifications.
* Minimal manual interference automated deployment pipeline.

Student Goals

* Complete testing code to industry standard.
* Plan, develop and manage an IT project.
* Expand their skills with real-world experience.
* Have a satisfied client in their work history.

Benefits

* Direct development to production deployment for rapid testing.
* Increase the speed of testing over manual testing.
* Releases become more reliable with fewer errors.
* Increased deployment efficiency.

# Methodology

Kanban was selected as the development methodology for this project as the dev team used it already to focus on their Continuous delivery (CD) and Continuous integration (CI) pipeline, which is essential for development teams to ship software fast while ensuring high quality.

Kanban is a method for managing workflow and controlling waste. A Kanban board visualises the workflow and monitors the project progress by showing tasks as cards that move across board swim lanes are they are progressed by the assignee.

The lead author implemented Kanban during the project to manage the WIP. Each of the six testing suite tasks was broken up into the sub-tasks:

* Creating e2e test scripts in Mocha to test if actions can be completed.
* Creating page objects using XPATH and Puppeteer to find UI elements and perform inputs.
* Creating page models and services to reference UI elements against backend API.

This process was repeated until all app functionalities had been tested, at which point the sub-task card could be moved from the in-progress lane of the Kanban Board to the testing lane. When all sub-tasks had been moved to the testing lane, a code review task was created and done between the author and a team member. When all the necessary changes had been refactored, the sub-task cards were moved to the done lane, and the main testing suite task was moved to the deployment lane. The app was deployed once the testing suite had been run and all passed in the CI/CD Pipeline. With this final step, the testing suite task card was moved to the done lane, and the task was considered complete and accepted, at which time the process began again with the next tested app.

# Project Management

The author selected the Agile Project Management Phases as the approach to plan and manage the project. Kanbanize (2023) explained that agile project management is an iterative process of managing requirements within a project. It focuses on flexibility, frequent value delivery, and fast feedback. The goal of using agile to manage your projects is an ability to adapt quickly and make changes when requirements change rapidly. Agile Project Management Phases consisted of the following:

* Envision – Identification of the objectives the client wishes the solution to achieve.
* Speculate - Requirements are determined, and a backlog of tasks is created with milestones and durations.
* Explore – Solutions to the requirements are explored as user stories are turned into tasks that make up deliverables that can be tested, iterated on, or released.
* Adapt – Deliverables, requirements, project approach, process, and objectives are adapted using customer and stakeholder feedback.
* Close - All deliverables are completed and released; reviews of the overall process are done so that lessons learned can be used to improve the next project.

# Risk management

The Risk Management process was completed weekly using the Microsoft (2022) suggested Microsoft risk template tool, which uses a scale of each risk’s probability and impact to rank all risks by the level of exposure they have. Along with the Clarkson (2012) suggested method of identifying potential risks using a What-if structure to brainstorm potential risks as a phase that begins with “what if…”.

# Quality Assurance

The Quality Assurance of this project’s deliverables was managed using a derivative of the Virginia Tech QA Table for management of Quality Assurance (2022). Each deliverable would go through the continuous improvement model of Plan, Do, Check, and Act to ensure feedback was implemented on each to keep the standards of all the deliverables high when they were accepted as complete by the stakeholder.

# Learnings

ICT-Specific Skills

* Author improved their technical skills in Typescript, testing, and version control.
* Author developed new skills in tools such as Xpath, Pipeline deployment, and API.

General Skills

* Author experienced Agile in a real-world environment.
* Author improved their communication and presentation skills in a professional workplace.
* Author gained experience in project management practices.

# Conclusion

Following the completion of the project, the lead author reflected on the successes and failures throughout the project. The lead author learnt a lot about development and project management during the project.

The main takeaway is that experience using tools like XPATH and Puppeteer in a real work environment taught the author the best practice of regression testing. While also learning from the challenging experience of struggling to debug team member’s code, there are still many skills left to learn and master.

During the project, the lead author learnt how important it is to be aware of and prioritise the work others are waiting for. Coming out of a situation where the author and a supervisor miscommunicated work priority. The lead author learnt to always ask about priorities in future work environments.

In conclusion, this project showed the author the possibilities and power of software automation. The speed at which a large testing suite can be run compared to manual testing is impressive and shows how it can be used to maintain overall app quality. The author has been fascinated with this technology and would like to see what things these tools could be used to automate.

# Referenc

Clarkson, J. (2012). *Beyond FMEA: the structured what-if technique (SWIFT)*. Retrieved from pubmed: https://www.researchgate.net/publication/224821158\_Beyond\_FMEA\_the\_structured\_what-if\_technique\_SWIFT

Microsoft. (2022, June 7). *Learn about insider risk management*. Retrieved from Microsoft.com: https://docs.microsoft.com/en-us/microsoft-365/compliance/insier-risk-management?view=o365-worldwide

Radigan, D. (2021). *What is kanban?* Retrieved March 17, 2023, from atlassian: https://www.atlassian.com/agile/kanban

Te Pūkenga. (2022, February 23). *A new kind of organisation*. Retrieved from tepūkenga: https://www.xn--tepkenga-szb.ac.nz/our-work/

Virginia Tech. (2022). *Project Quality Assurance*. Retrieved from Virginia tech: https://it.vt.edu/projects/project\_management/qualityassurance.html

W3C consortium. (2018). *Web Content Accessibility Guidelines (WCAG).* Retrieved from Web Accessibility Initiative: http://www.w3.org/WAI/intro/wcag.php