QA Intern Report

Automated Signup Flow with OTP Verification

Intern Name: Santosh Kumar Karki **Date:** 2026/01/01

1 Introduction

This report documents an automated end-to-end signup testing script developed as part of a QA internship task. The automation validates a multi-step registration workflow that includes form submission, dropdown selections, file uploads, and OTP-based email verification. The goal is to reduce repetitive manual testing while ensuring functional correctness of the signup process.

2 Objectives

- Automate the complete signup workflow of the web application
- Verify OTP-based email validation using a temporary mailbox
- Handle dynamic UI components built with React/Radix UI
- Demonstrate reliable QA automation practices

3 Tools and Environment

- **Language:** Python 3.11.8
- **Automation Tool:** Selenium WebDriver
- **Browser:** Google Chrome
- **API Service:** mail.tm (temporary email & OTP retrieval)
- **Libraries:** requests, names, selenium

4 High-Level Test Flow

1. Launch application and navigate to Register page
2. Accept terms and proceed to account setup
3. Generate temporary email using mail.tm API
4. Fill personal details and submit the form
5. Retrieve OTP from email inbox and verify account
6. Complete agency, experience, and verification forms
7. Upload required documents
8. Submit final registration

5 Automation Design

OTP Handling

The application uses a single hidden input to manage OTP entry. The automation directly targets this input using a stable `data-*` attribute and inputs the extracted OTP digits programmatically.

Dynamic UI Handling

Explicit waits (`WebDriverWait`) are used to manage dynamic DOM updates. A retry mechanism handles stale element exceptions commonly seen in SPA applications.

Locator Strategy

Preference is given to:

- `name` and `id` attributes
- `data-*` attributes
- Text-based XPath selectors

This approach improves script stability and maintainability.

6 Results

The automation successfully completes the signup process without manual intervention. OTP verification is handled reliably, forms are submitted correctly, and required documents are uploaded. The script demonstrates consistent execution across multiple test runs.

7 Limitations and Future Improvements

- OTP delivery delays may affect execution time
- No negative test cases implemented
- No assertions or screenshot capture on failure

Future enhancements include adding assertions, logging, screenshot capture, and converting the script into a PyTest-based framework with Page Object Model (POM).

8 Conclusion

This automation script demonstrates a complete and realistic QA automation scenario, covering UI interaction, API integration, and OTP validation. The implementation reflects industry-standard practices suitable for a QA internship and can be extended for regression testing with minimal effort.