CleanCity Waste Pickup Scheduler – QA Test Report

Version: 1.0.0

Tester: Obed Magori

Date: 24 Jul 2025

1. Summary

This QA report covers the functional, UI/UX, and security testing of the CleanCity Waste Pickup Scheduler web application. The purpose of the test was to identify functional gaps, potential vulnerabilities, and user experience issues before production deployment.

2. Environment

Platform	Web (React Single-Page Application)
Browsers Tested	Chrome 138, Firefox 128
Operating System	Windows 10
Tools Used	Chrome DevTools (Elements, Console, Network, Application), Lighthouse Audit

3. Test Cases

ID	Description	Expected Result	Actual Result	Status
TC-001	Login with valid credentials (demo user)	User should be redirected to dashboard	No API call found, login not functional	Fail
TC-002	Register new user	Account created and stored	No persistence, only UI confirmation	Fail
TC-003	Pickup request form (missing name)	Show error message	Error message displayed	Pass
TC-004	Pickup request date validation	Reject past dates	Past dates accepted	Fail
TC-005	Toggle mobile menu	Menu opens and closes	Unable to confirm without JS	Needs Retest
TC-006	Reveal admin link via DevTools	Server should block access	Admin link accessible	Fail
TC-007	Inspect local storage	No sensitive data stored	No credentials found	Pass

4. Critical Bugs

- **BUG-001**: No backend authentication – all login validation is front-end only.
- **BUG-002**: Weak password policy – accepts passwords with less than 8 characters.
- **BUG-003**: Admin link exposure – can be revealed and accessed via DevTools.
- **BUG-004**: Pickup date accepts past dates – no validation applied.
- **BUG-005**: Hardcoded demo credentials visible in HTML source.

5. Recommendations

- Implement secure backend API for authentication, registration, and data persistence.
- Strengthen password requirements (min 8 characters, mixed case, numbers, special symbols).
- Apply server-side role validation for admin access; remove reliance on UI hiding.
- Validate dates on both client and server to prevent past-date entries.
- Remove hardcoded credentials from HTML source.
- Add CAPTCHA or rate-limiting to prevent automated spam submissions.
- Provide loading indicators during form submissions for better UX.
- Perform complete responsive UI testing once JS files are fully integrated.