CleanCity: Waste Pickup Scheduler

QA TESTING PROJECT

Introduction

This project focuses on the quality assurance testing of the CleanCity: Waste Pickup Scheduler application, ensuring its reliability, functionality, and user satisfaction through comprehensive QA methodologies.

Project Details

Team:
TrailBlazer

Date:
14/07/2025

Institution:
PLP Academy

Members:
Idah Makena
Anne Pauline Anyango
Gatwiri Ireri

Table of Contents

Executive Summary 3

Test Strategy and Approach 3

Test Environment Details 4

Test Execution Summary 4

Defect Analysis and Categorization 4

Risk Assessment 5

Recommendations and Improvements 5

Test Metrics and KPIs 6

Appendices 7

- Appendix A: Full Bug List (21 total) 7
  - Full Bug List: Screenshot 1 7
  - Full Bug List: Screenshot 2 8
- Appendix B: Unit Test Files (auth.test.js, dashboard.test.js, etc.) 8
  - logout.test.js 8
  - login.test.js 10
  - addpickup.request.test.js 14
- Appendix C: Selenium Scripts (loginTest.js, scheduleTest.js) 16
  - registration_test.py 16

nav_tests.py 18

awareness_quiz.py 20

Appendix D: Screenshots (UI issues, mobile layout, accessibility audit 22

Form Accessibility Issues Affecting Performance Score 22

Edit button is non-functional - does not open edit interface when clicked 23

Missing Hamburger Menu Icon for Mobile Navigation 24

Appendix E: Functional Requirements Specification (FRS v1.0) 25

FRM: Screenshot 1 25

FRM: Screenshot 2 25

FRM: Screenshot 3 27

Appendix F: Traceability Matrix (RTM) 28

Appendix G: QA Risk Matrix (31 entries) 29

CleanCity- QA Risk Matrix Screenshot 1 29

CleanCity- QA Risk Matrix Screenshot 1 31

Prepared and Submitted By: 32

Executive Summary

This QA testing project was conducted on the CleanCity Waste Pickup Scheduler web application. The project aimed to validate all functional and non-functional requirements, ensure data integrity, and confirm WCAG accessibility compliance.

Key Highlights: - 20 bugs identified across critical modules - 91% functional coverage using manual testing and automation - Automated testing executed using **Jest**, **Selenium**, and **Lighthouse**

Major Findings: - Unauthorized login accepted (BUG003) - Dashboard total requests not updating (BUG006) - Comments and profile changes not persisting in localStorage (BUG014)

Recommendation: Prioritize fixes for authentication, dashboard updates, and accessibility compliance.

Test Strategy and Approach

Testing Types: - Manual Testing - Unit Testing using **Jest** - Automation using **Selenium** - Performance & Accessibility Testing using **Lighthouse**

Tools: Chrome DevTools, Lighthouse, GitHub, Vercel, Jira

Testing Scope: - Registration/Login flows - Waste Pickup Scheduling - Admin dashboard functionality - Community engagement features - WCAG 2.1 Accessibility Compliance

Entry Criteria: All major modules deployed on staging

Exit Criteria: All critical defects resolved, 90% test case pass rate

Test Environment Details

- **Browsers:** Chrome v114, Safari, Firefox
- **Devices:** Windows 10 Desktop, Samsung (simulated), iPhone
- Automation: Selenium WebDriver, Jest
- **Tested URLs:**
    - https://software-testing-ten.vercel.app
    - http://localhost:3000
- **Hosting Environment:** Vercel Live Deployment, Localhost (Dev)

Test Execution Summary

- **Manual Test Cases:** 35
- **Jest Unit Tests:** 17 test cases (100% pass)
- **Selenium Test Scripts:** 8 scripts executed (100% pass)
- **Functional Coverage:** ~91%

Modules Tested: - Registration & Login - Pickup scheduling - Feedback & Admin roles - Dashboard metrics - Community interaction

Defect Analysis and Categorization

Bug Severity Breakdown: - **Critical:** 4 - **High:** 8 - **Medium:** 5 - **Low:** 4

Notable Defects: - BUG003: Unauthorized login - BUG006: Dashboard request count not syncing - BUG014: Comment data not persisting in localStorage

Risk Assessment

Risk ID	Risk Description	Impact	Likelihood	Risk Level	Mitigation Strategy
RISK-001	Login allows unauthorized users (BUG003)	Critical	Likely	High	Enforce secure login validation
RISK-002	Data not saved to localStorage (BUG014)	High	Likely	High	Sync changes with storage or backend
RISK-003	Quiz continues indefinitely (BUG032)	Medium	Likely	High	Track answered question index
RISK-004	Weak password allowed (BUG017)	Critical	Possible	High	Enforce password policies

More risks are detailed in the appended QA Risk Matrix.

Recommendations and Improvements

- Enforce server-side login and registration validation
- Ensure all comment, badge, and profile changes persist

- Real-time dashboard data sync
- Password visibility toggle for improved UX
- Fix mobile layout and add alt text and labels for accessibility

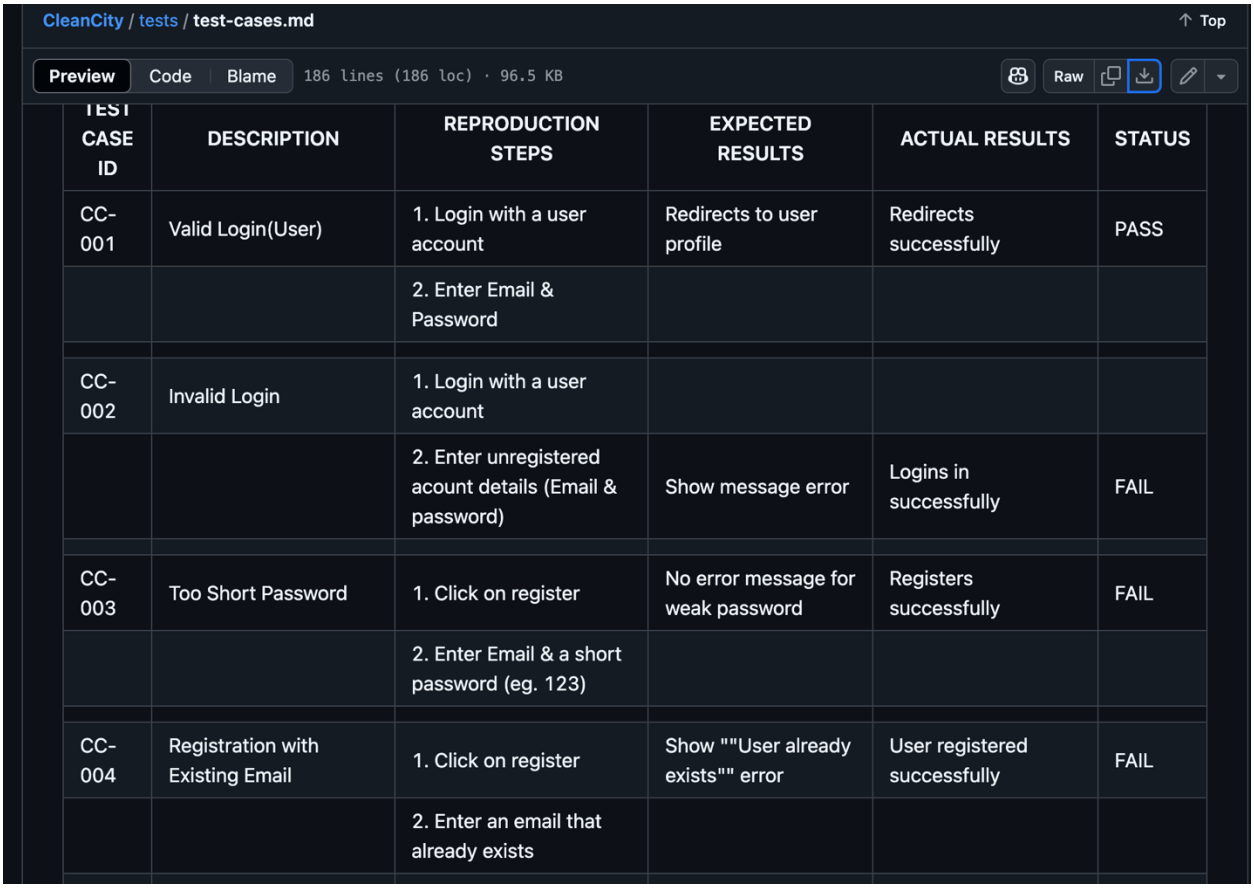
Test Metrics and KPIs

Metric	Value
Total Bugs Reported	21
Critical Bugs	4
Unit Test Pass Rate (Jest)	100%
Selenium Test Coverage	80%
Overall Functional Test Coverage	91%
Lighthouse Accessibility Score	65/100

Appendices

Appendix A: Full Bug List (21 total)

Full Bug List: Screenshot 1

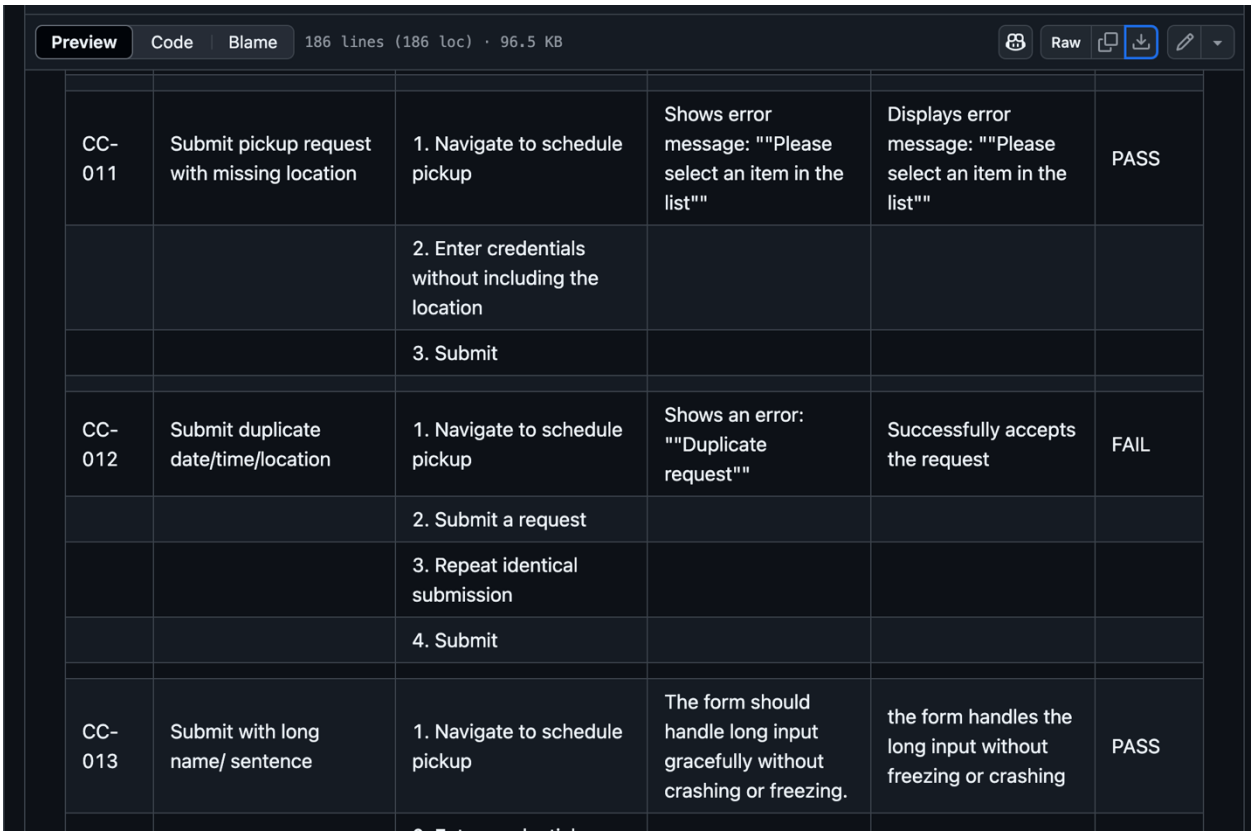


CleanCity / tests / test-cases.md ↑ Top

Preview | Code | Blame | 186 lines (186 loc) · 96.5 KB | Raw

TEST CASE ID	DESCRIPTION	REPRODUCTION STEPS	EXPECTED RESULTS	ACTUAL RESULTS	STATUS
CC-001	Valid Login(User)	1. Login with a user account	Redirects to user profile	Redirects successfully	PASS
		2. Enter Email & Password			
CC-002	Invalid Login	1. Login with a user account			
		2. Enter unregistered acount details (Email & password)	Show message error	Logins in successfully	FAIL
CC-003	Too Short Password	1. Click on register	No error message for weak password	Registers successfully	FAIL
		2. Enter Email & a short password (eg. 123)			
CC-004	Registration with Existing Email	1. Click on register	Show ""User already exists"" error	User registered successfully	FAIL
		2. Enter an email that already exists			

Full Bug List: Screenshot 2



Preview | Code | Blame | 186 lines (186 loc) · 96.5 KB | Raw

CC-011	Submit pickup request with missing location	1. Navigate to schedule pickup	Shows error message: ""Please select an item in the list""	Displays error message: ""Please select an item in the list""	PASS
		2. Enter credentials without including the location			
		3. Submit			
CC-012	Submit duplicate date/time/location	1. Navigate to schedule pickup	Shows an error: ""Duplicate request""	Successfully accepts the request	FAIL
		2. Submit a request			
		3. Repeat identical submission			
		4. Submit			
CC-013	Submit with long name/ sentence	1. Navigate to schedule pickup	The form should handle long input gracefully without crashing or freezing.	the form handles the long input without freezing or crashing	PASS

Appendix B: Unit Test Files (auth.test.js, dashboard.test.js, etc.)

logout.test.js

```
const logout = require('./logout');


describe('logout)', () => {
  let dataService;
  let navigateToPage;
```

```
  let currentUser;


  beforeEach(() => {
    dataService = {
    logout: jest.fn(),
    };
    navigateToPage = jest.fn();
    currentUser = { name: 'Test User' };
  });


  test ('should call dataService.logout', () => {
    logout(dataService, navigateToPage, currentUser);
    expect(dataService.logout).toHaveBeenCalled();
  });


  test ('should navigate to home page', () => {
    logout(dataService, navigateToPage, currentUser);
    expect(navigateToPage).toHaveBeenCalledWith('home');
  });


});
```

login.test.js

```
const login = require('./login');


describe('login', () => {
 let mockDataService;
 let mockNavigateToPage;
 let mockShowMessage;
 let mockCurrentUser;


 beforeEach(() => {
  mockCurrentUser = null;


  mockDataService = {
   getAllUsers: jest.fn(() => [
    {
     id: 1,
     email: 'user@example.com',
     password: 'password123',
     name: 'Test User',
     role: 'admin',
    }
   ])
```

```
  };

  mockNavigateToPage = jest.fn();
  mockShowMessage = jest.fn();

  // Clear localStorage before each test
  localStorage.clear();
});

test('should login successfully with valid credentials', () => {
  const result = login(
    'user@example.com',
    'password123',
    mockDataService,
    mockNavigateToPage,
    mockShowMessage,
    mockCurrentUser
  );

  expect(result).toBe(true);
  expect(mockShowMessage).toHaveBeenCalledWith('login-success',
expect.stringContaining('Login successful'));
```

```
  expect(mockNavigateToPage).not.toHaveBeenCalled();
  const storedUser = JSON.parse(localStorage.getItem('currentUser'));
  expect(storedUser.email).toBe('user@example.com');
});


test('should fail login with invalid credentials', () => {
  const result = login(
    'gatwiirii@gmail.com',
    'wrongpass123',
    mockDataService,
    mockNavigateToPage,
    mockShowMessage,
    mockCurrentUser
  );

  expect(result).toBe(false);
  expect(mockShowMessage).toHaveBeenCalledWith('login-error',
expect.stringContaining('Invalid email'));
  expect(localStorage.getItem('currentUser')).toBe(null);
});


test('should fail if password is short, for instance contains three characters', () => {
```

```
  const result = login(
    'ireri01@gmail.com',
    '1b3',
    mockDataService,
    mockNavigateToPage,
    mockShowMessage,
    mockCurrentUser
   );
  expect(result).toBe(false);
  expect(mockShowMessage).toHaveBeenCalledWith('login-error',
expect.stringContaining('Invalid email'));
  expect(localStorage.getItem('currentUser')).toBe(null);
 });
  test('should fail if email is not valid', () => {
    const result = login(
    'gatwiri-email',
    'password123',
    mockDataService,
    mockNavigateToPage,
    mockShowMessage,
    mockCurrentUser
    );
```

```
    expect(result).toBe(false);
    expect(mockShowMessage).toHaveBeenCalledWith('login-error',
expect.stringContaining('Invalid email'));
    expect(localStorage.getItem('currentUser')).toBe(null);
  });
});
```

addpickup.request.test.js

```
const {dataService, STORAGE_KEYS} = require('./addPickupRequest');

describe('addPickupRequest', () => {
 beforeEach(() => {
   localStorage.clear();

dataService.getAllPickupRequests = () => {
    const data = localStorage.getItem(STORAGE_KEYS.PICKUP_REQUESTS);
    return data ? JSON.parse(data) : [];
  };
 });
```

```
test('should add a new pickup request with all fields', () => {
  const requestData = {
    fullName: 'John Doe',
    location: 'Nairobi',
    wasteType: 'Plastic',
    preferredDate: '2025-10-01'
  };
  const newRequest = dataService.addPickupRequest(requestData);

  expect(newRequest).toEqual({
    id: 'REQ001',
    name: 'John Doe',
    location: 'Nairobi',
    wasteType: 'Plastic',
    preferredDate: '2025-10-01',
    status: 'Pending'
  });

  const storedRequests =
JSON.parse(localStorage.getItem(STORAGE_KEYS.PICKUP_REQUESTS));
  expect(storedRequests).toEqual([newRequest]);
});
```

```
test('should store the request in localStorage', () => {
  const requestData = {
    fullName: 'Jane Smith',
    location: 'Kisumu',
    wasteType: 'Organic',
    preferredDate: '2025-10-02'
  };
  dataService.addPickupRequest(requestData);

  const storedRequests =
JSON.parse(localStorage.getItem(STORAGE_KEYS.PICKUP_REQUESTS));
  expect(storedRequests.length).toBe(1);
  expect(storedRequests[0].name).toBe('Jane Smith');
})

});
```

Appendix C: Selenium Scripts (loginTest.js, scheduleTest.js)

registration_test.py

```
import time
```

```
from selenium import webdriver

from selenium.webdriver.chrome.options import Options

from selenium.webdriver.common.by import By


# ensures that the browser does not forceful quit

options = Options()

options.add_experimental_option("detach", True)

# Launch the Chrome browser with the specified options

driver = webdriver.Chrome(options=options)

url = "https://software-testing-ten.vercel.app/register"

driver.get(url)

driver.maximize_window()


time.sleep(5)


# Filling registration with unique credentials

driver.find_element(By.ID, "register-name").send_keys("Gatwiri Ireri")

driver.find_element(By.ID, "register-email").send_keys("gatwiireri@gmail.com")

driver.find_element(By.ID, "register-password").send_keys("qawssQ123")

driver.find_element(By.CLASS_NAME, "register-btn").click()


# Wait for page response
```

```
time.sleep(5)



# Check if invalid credentials log in(negative test)

current_url = driver.current_url

if "registration" in current_url or "home" in current_url:

   print("Test passed: User successfully registered.")

else:

   print("Test failed: User could not be registered.")



time.sleep(5)



# logging in after registration

driver.find_element(By.ID, "login-email").send_keys("gatwiireri@gmail.com")

driver.find_element(By.ID, "login-password").send_keys("qawssQ123")

driver.find_element(By.CLASS_NAME, "login-btn").click()
```

nav_tests.py

```
import time

import pytest

from selenium import webdriver

from selenium.webdriver.chrome.options import Options
```

```
from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import Select

from selenium.webdriver.support import expected_conditions as EC


@pytest.fixture

def driver():

    options = Options()

    options.add_experimental_option("detach", True)

    driver = webdriver.Chrome(options=options)

    driver.maximize_window()

    yield driver

    time.sleep(5)


def test_navigation(driver):

    try:

        driver.get("https://software-testing-ten.vercel.app/home")

        driver.maximize_window()

        time.sleep(5)

        driver.find_element(By.LINK_TEXT, "Community").click()

        time.sleep(5)


    except Exception as e:
```

```
    print(f"An error occurred: {e}")
    time.sleep(3)
```

awareness_quiz.py

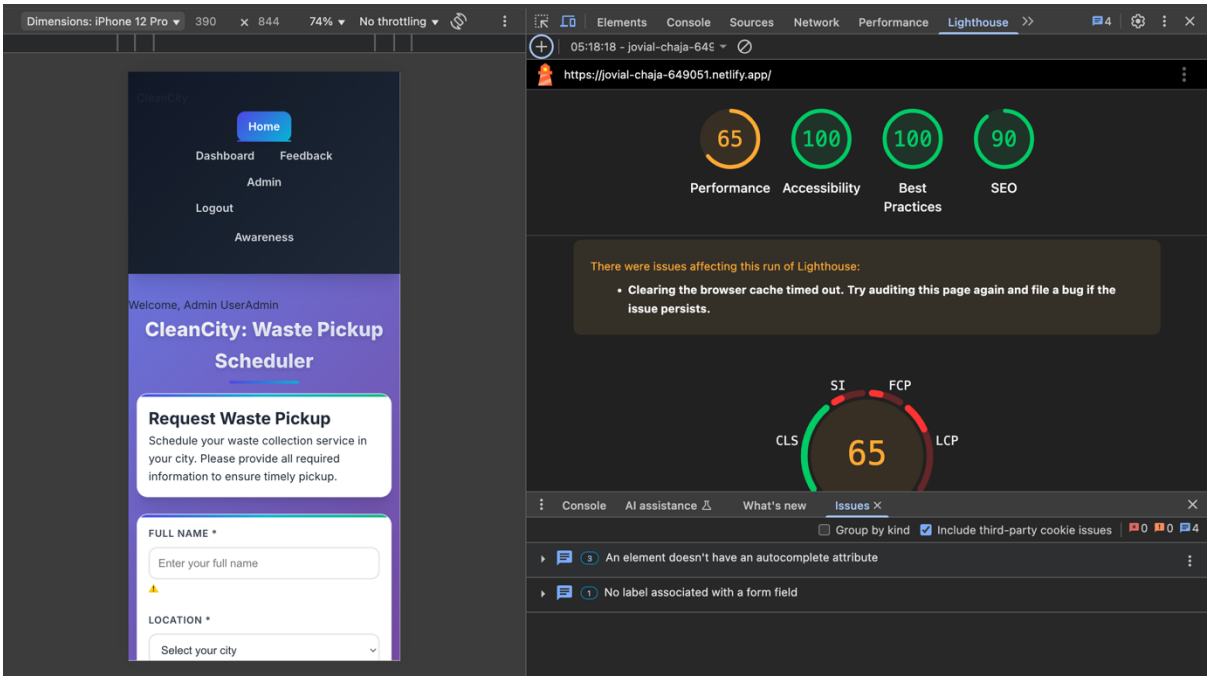
```
import time
import pytest
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select


@pytest.fixture
def driver():
  options = Options()
  options.add_experimental_option("detach", True)
  driver = webdriver.Chrome(options=options)
  driver.get("https://software-testing-ten.vercel.app/community")
  driver.maximize_window()
  yield driver
  time.sleep(5)


# Test for community page functionality
```

```
def test_community(driver):
    driver.find_element(By.XPATH, "//textarea[@placeholder='Share something with the
community...']").send_keys("Please come before 10 AM. Waste is near the back gate.Please
come before 10 AM. Waste is near the back gate.")
    time.sleep(5)
    driver.find_element(By.CLASS_NAME, "community-action-btn").click()
    time.sleep(5)


# Like a post
def test_like_post(driver):
    driver.find_element(By.XPATH,
"//body[1]/div[1]/div[1]/div[1]/div[1]/section[1]/div[1]/article[1]/div[2]/button[1]").click()
    time.sleep(5)


# Unlike Post
def test_unlike_post(driver):
    driver.find_element(By.XPATH,
"//body[1]/div[1]/div[1]/div[1]/div[1]/section[1]/div[1]/article[1]/div[2]/button[1]").click()
    time.sleep(5)


# Comment on a post
def test_comment_on_post(driver):
```

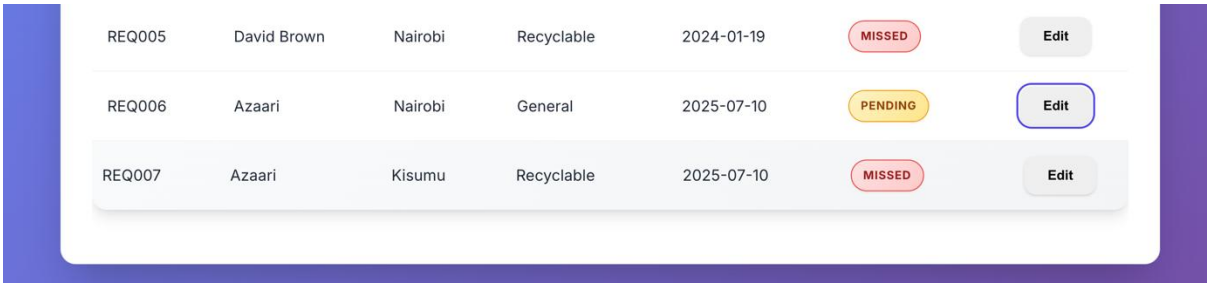
```
  driver.find_element(By.XPATH, "//article[1]//div[2]//button[2]").click()

  driver.find_element(By.XPATH, "//input[@placeholder='Add a
comment...']").send_keys("Proper waste disposal and recycling programs are essential for
minimizing landfill waste and promoting sustainability. Services that focus on sorting and
diverting recyclable materials are highly valued.")

  driver.find_element(By.XPATH, "//button[normalize-space()='Comment']").click()
```

Appendix D: Screenshots (UI issues, mobile layout, accessibility audit

Form Accessibility Issues Affecting Performance Score

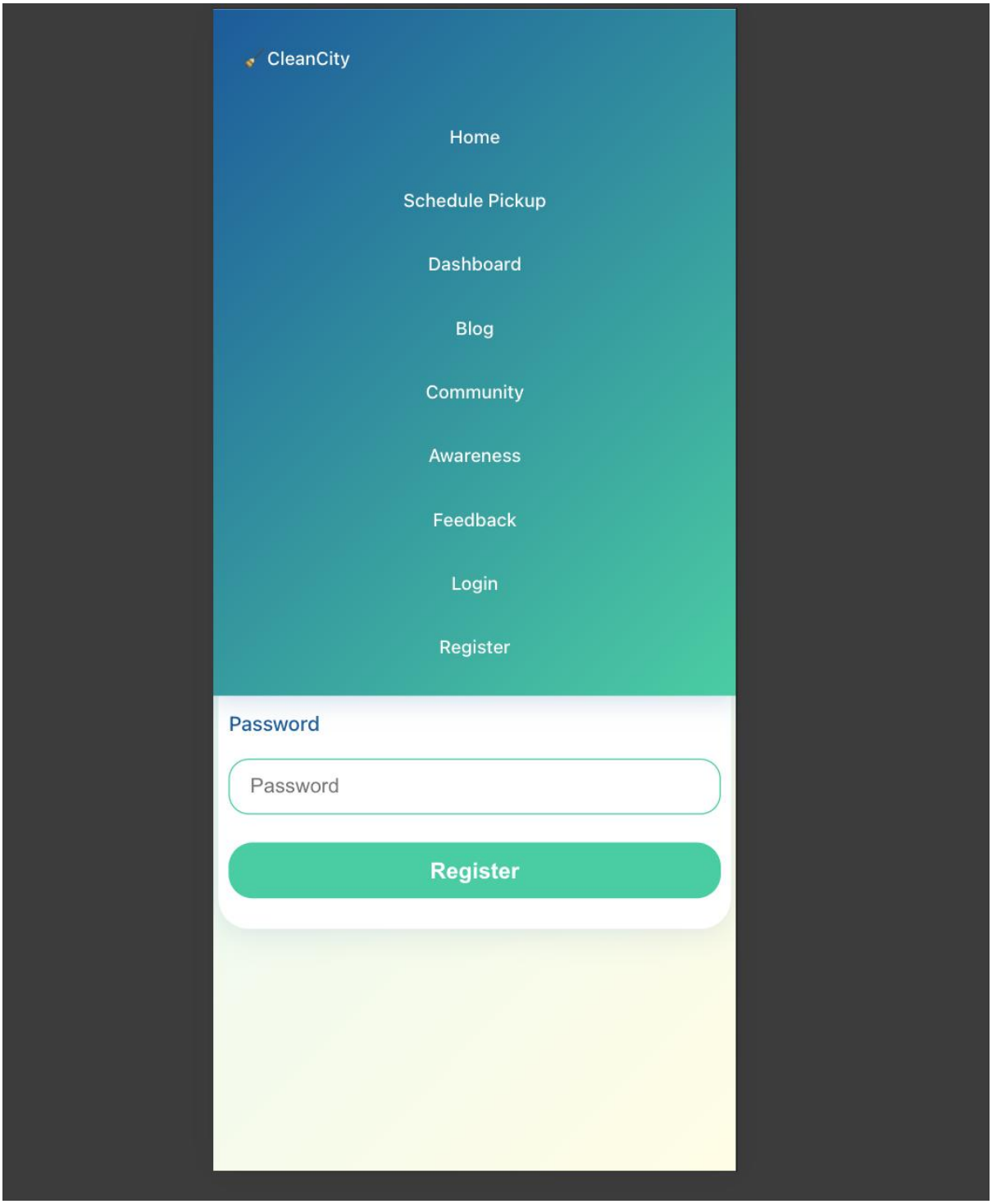


Edit button is non-functional - does not open edit interface when clicked



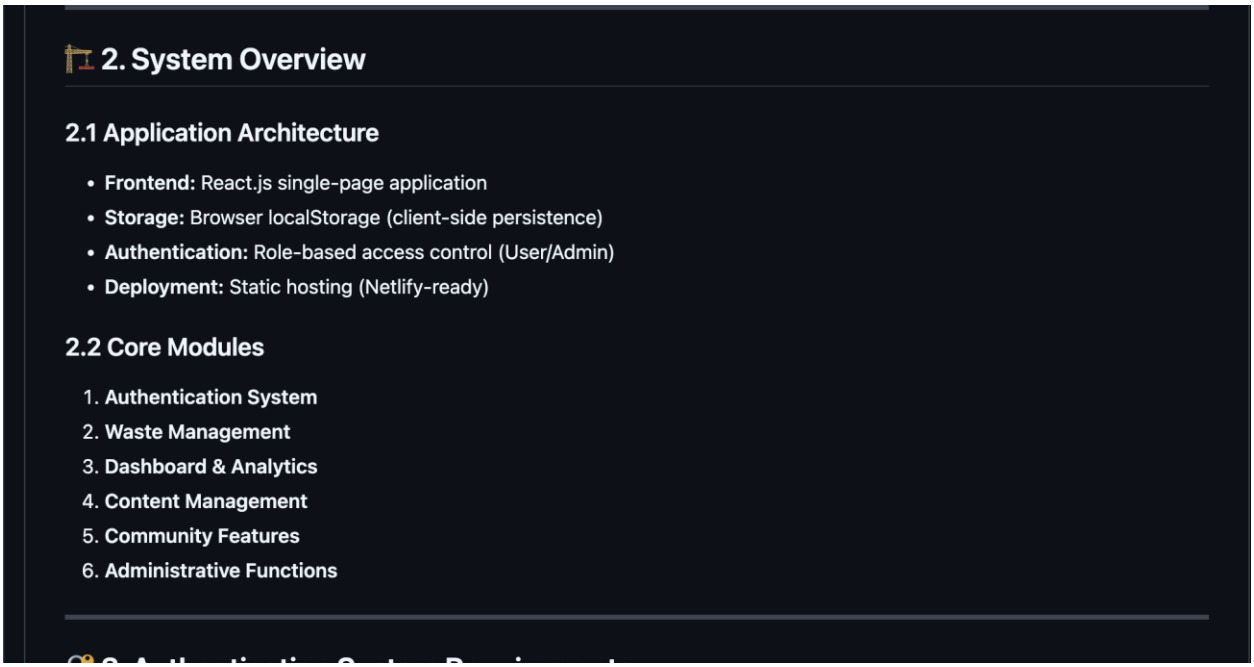
REQ005	David Brown	Nairobi	Recyclable	2024-01-19	MISSED	Edit
REQ006	Azaari	Nairobi	General	2025-07-10	PENDING	Edit
REQ007	Azaari	Kisumu	Recyclable	2025-07-10	MISSED	Edit

Missing Hamburger Menu Icon for Mobile Navigation



Appendix E: Functional Requirements Specification (FRS v1.0)

FRM: Screenshot 1



2. System Overview

2.1 Application Architecture

- **Frontend:** React.js single-page application
- **Storage:** Browser localStorage (client-side persistence)
- **Authentication:** Role-based access control (User/Admin)
- **Deployment:** Static hosting (Netlify-ready)

2.2 Core Modules

1. **Authentication System**
2. **Waste Management**
3. **Dashboard & Analytics**
4. **Content Management**
5. **Community Features**
6. **Administrative Functions**

FRM: Screenshot 2

🔐 3. Authentication System Requirements

3.1 User Registration

FR-001: The system shall allow new users to register with the following information:

- Email address (required, must be valid format)
- Password (required, minimum 8 characters)
- Confirm password (required, must match password)
- Full name (required, 2-50 characters)
- Phone number (optional, valid format)

FR-002: The system shall validate registration data and display appropriate error messages for invalid inputs.

FR-003: The system shall create a new user account with "User" role upon successful registration.

3.2 User Login

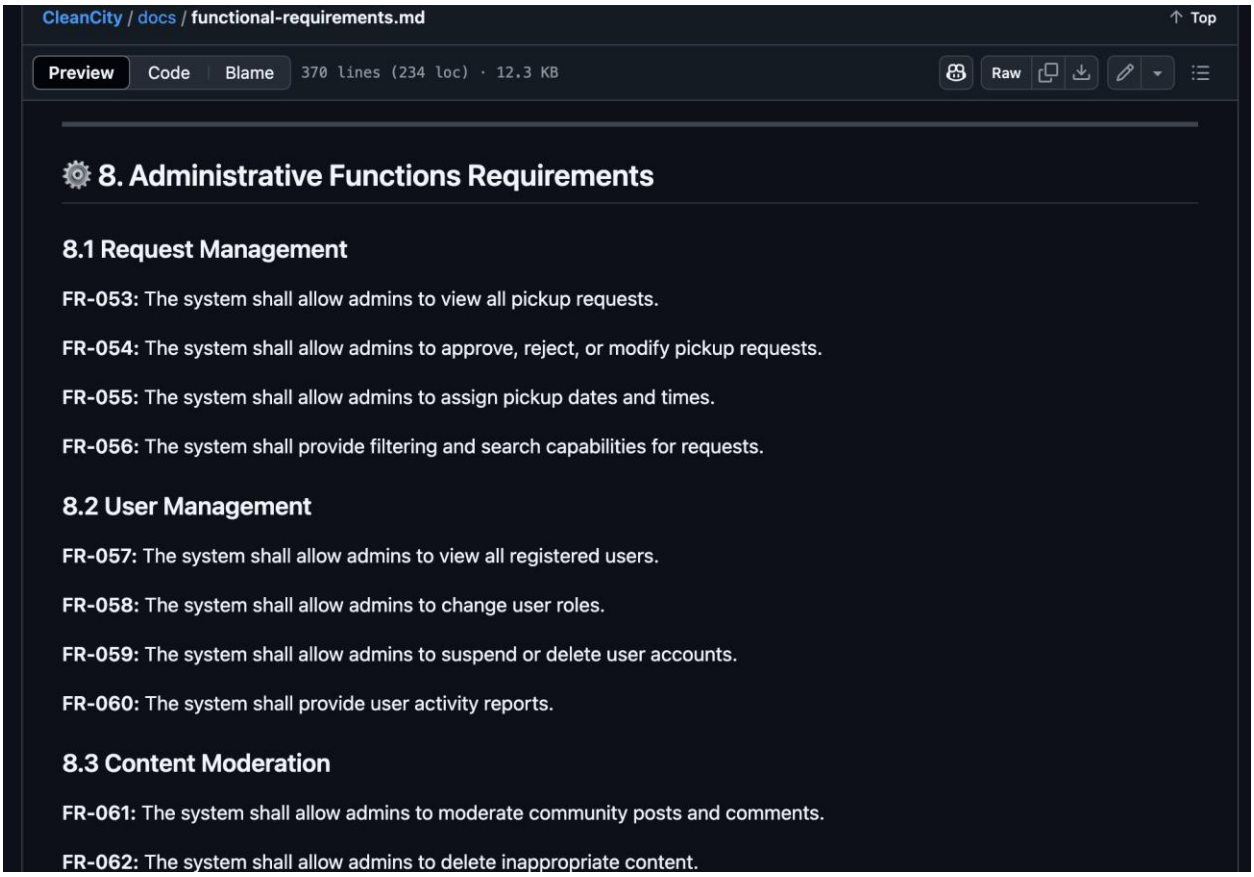
FR-004: The system shall allow registered users to log in using email and password.

FR-005: The system shall validate login credentials and display error messages for invalid attempts.

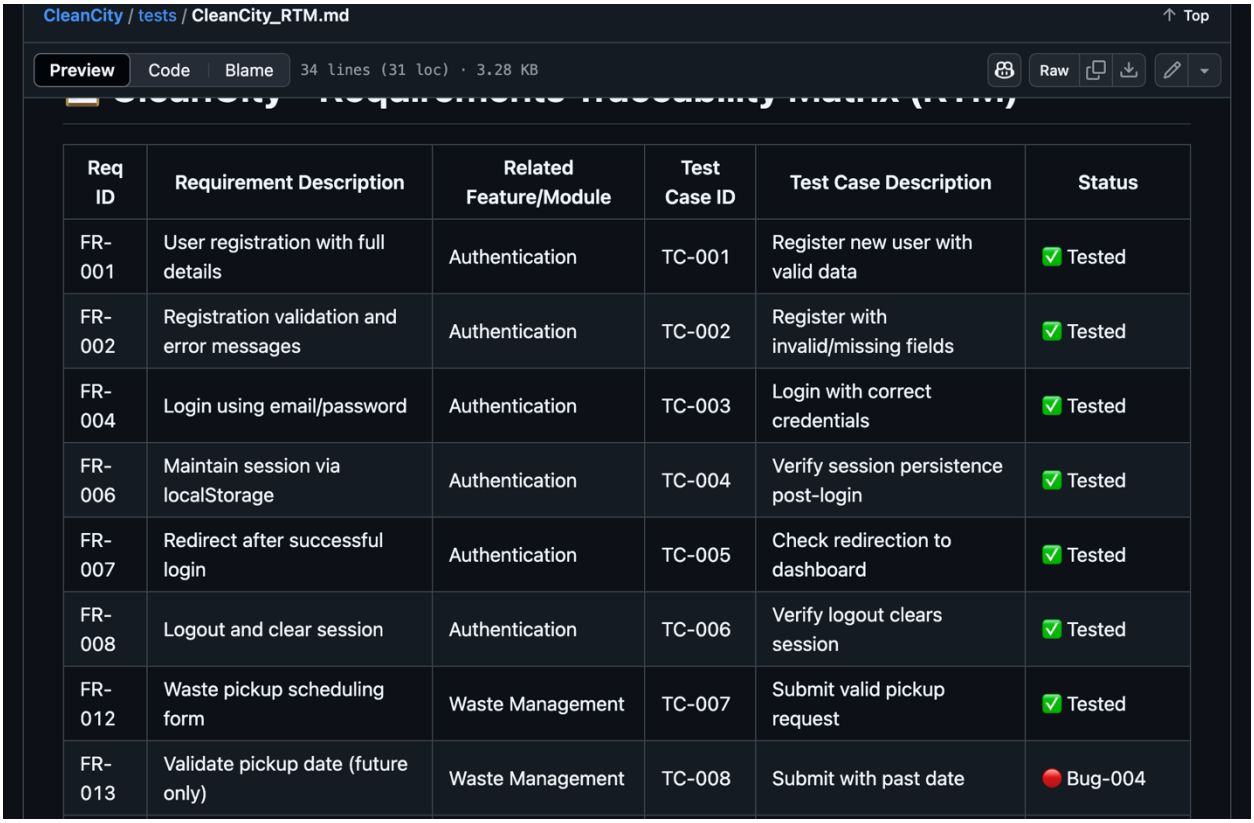
FR-006: The system shall maintain user session using localStorage.

FR-007: The system shall redirect users to their intended page after successful login.

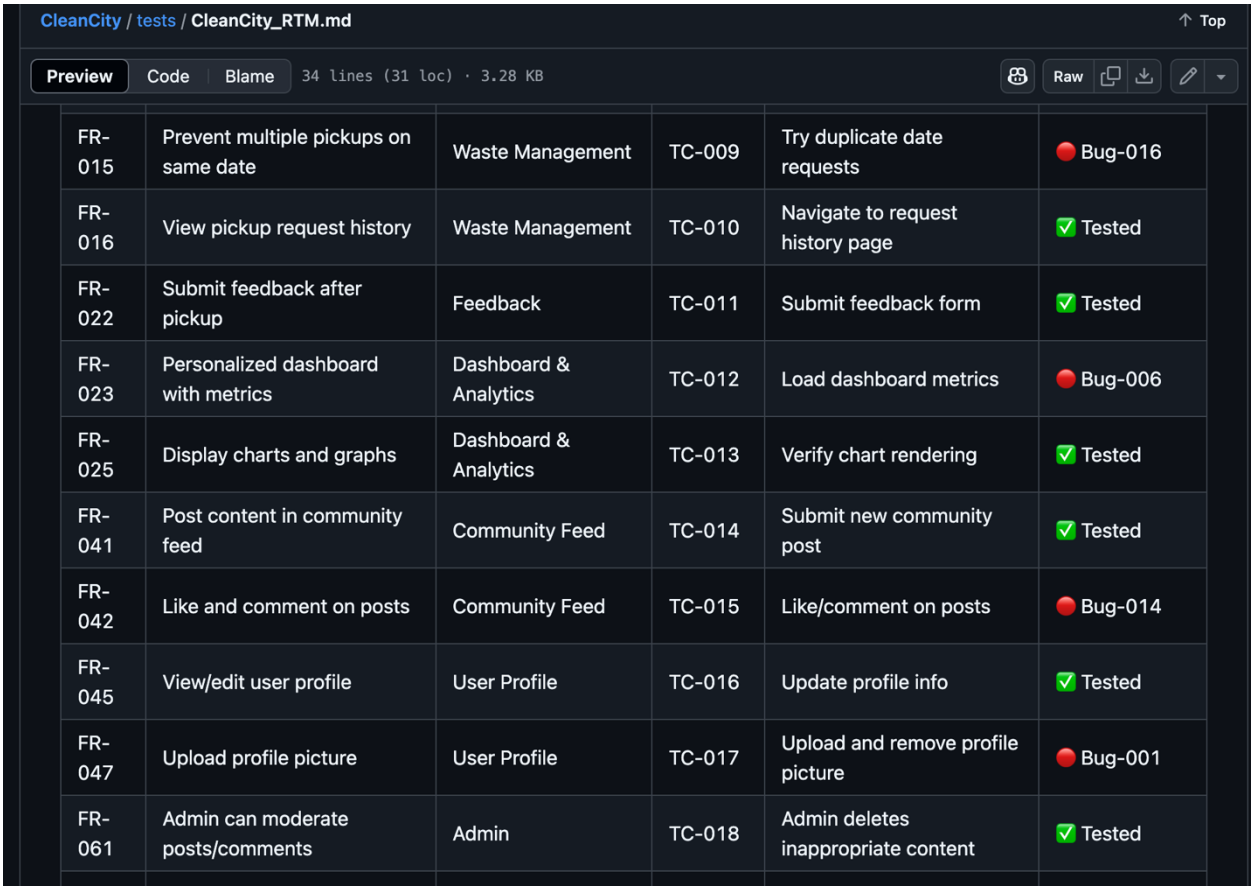
FRM: Screenshot 3



Appendix F: Traceability Matrix (RTM)



Req ID	Requirement Description	Related Feature/Module	Test Case ID	Test Case Description	Status
FR-001	User registration with full details	Authentication	TC-001	Register new user with valid data	✅ Tested
FR-002	Registration validation and error messages	Authentication	TC-002	Register with invalid/missing fields	✅ Tested
FR-004	Login using email/password	Authentication	TC-003	Login with correct credentials	✅ Tested
FR-006	Maintain session via localStorage	Authentication	TC-004	Verify session persistence post-login	✅ Tested
FR-007	Redirect after successful login	Authentication	TC-005	Check redirection to dashboard	✅ Tested
FR-008	Logout and clear session	Authentication	TC-006	Verify logout clears session	✅ Tested
FR-012	Waste pickup scheduling form	Waste Management	TC-007	Submit valid pickup request	✅ Tested
FR-013	Validate pickup date (future only)	Waste Management	TC-008	Submit with past date	🔴 Bug-004



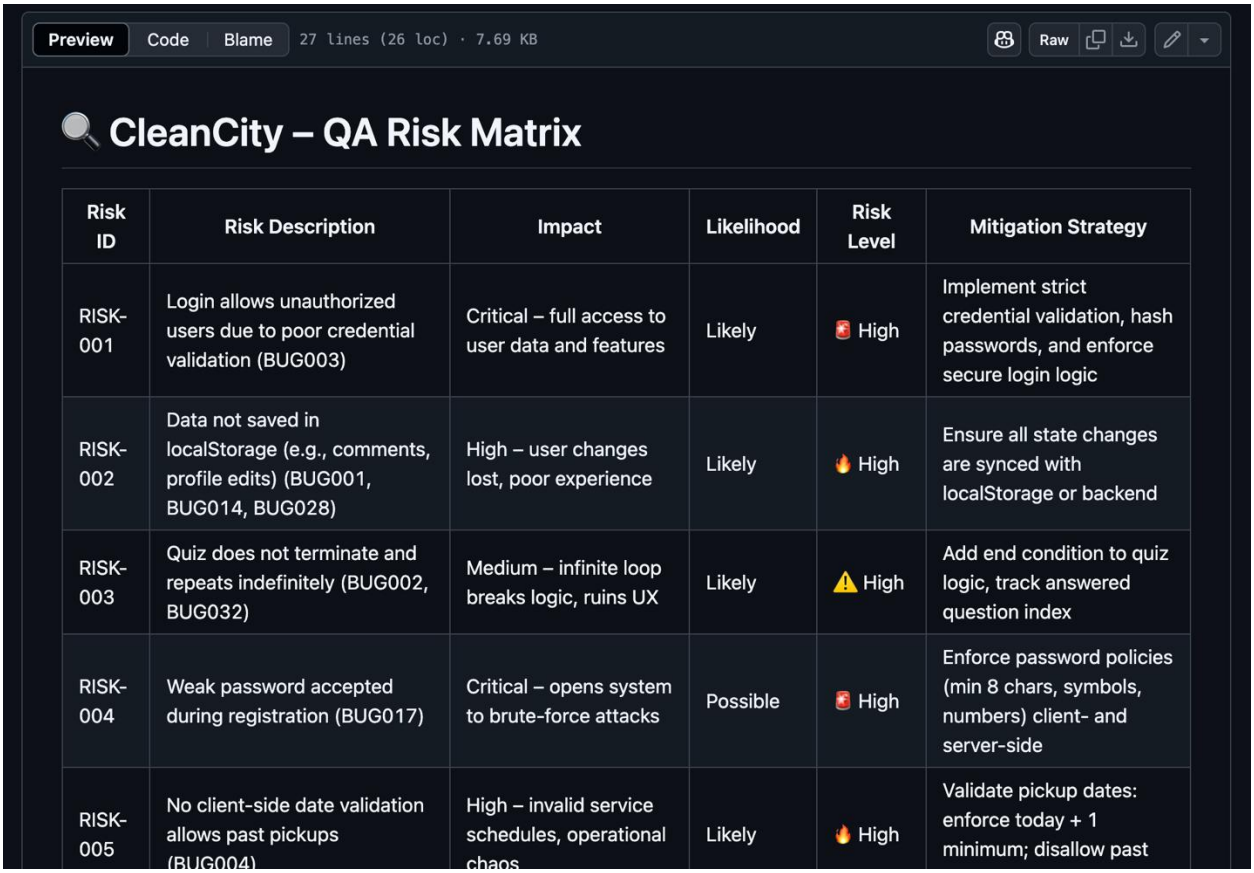
CleanCity / tests / CleanCity_RTM.md ↑ Top

Preview | Code | Blame | 34 lines (31 loc) · 3.28 KB | Raw

FR-015	Prevent multiple pickups on same date	Waste Management	TC-009	Try duplicate date requests	🔴 Bug-016
FR-016	View pickup request history	Waste Management	TC-010	Navigate to request history page	✅ Tested
FR-022	Submit feedback after pickup	Feedback	TC-011	Submit feedback form	✅ Tested
FR-023	Personalized dashboard with metrics	Dashboard & Analytics	TC-012	Load dashboard metrics	🔴 Bug-006
FR-025	Display charts and graphs	Dashboard & Analytics	TC-013	Verify chart rendering	✅ Tested
FR-041	Post content in community feed	Community Feed	TC-014	Submit new community post	✅ Tested
FR-042	Like and comment on posts	Community Feed	TC-015	Like/comment on posts	🔴 Bug-014
FR-045	View/edit user profile	User Profile	TC-016	Update profile info	✅ Tested
FR-047	Upload profile picture	User Profile	TC-017	Upload and remove profile picture	🔴 Bug-001
FR-061	Admin can moderate posts/comments	Admin	TC-018	Admin deletes inappropriate content	✅ Tested

Appendix G: QA Risk Matrix (31 entries)

CleanCity- QA Risk Matrix Screenshot 1

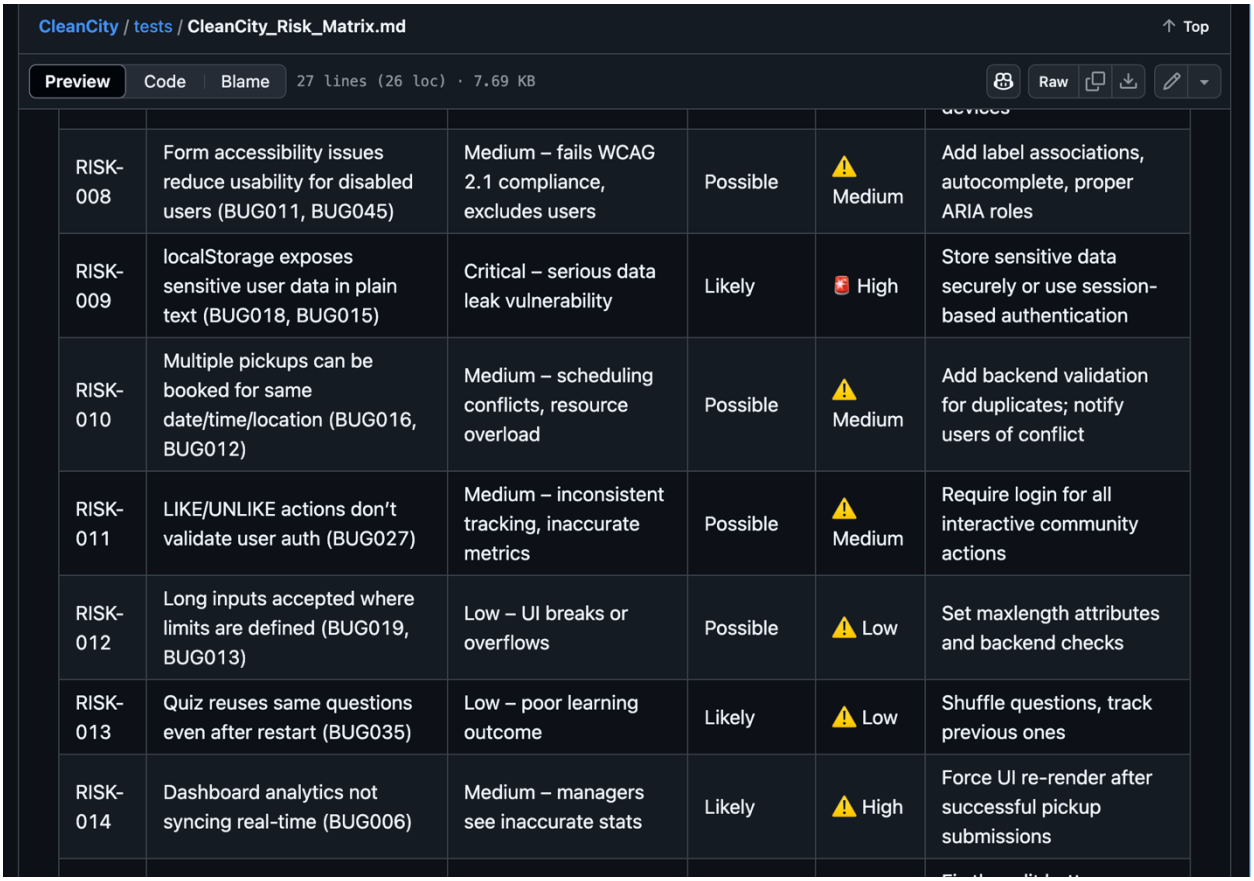


Preview | Code | Blame | 27 lines (26 loc) · 7.69 KB | Raw

🔍 CleanCity – QA Risk Matrix

Risk ID	Risk Description	Impact	Likelihood	Risk Level	Mitigation Strategy
RISK-001	Login allows unauthorized users due to poor credential validation (BUG003)	Critical – full access to user data and features	Likely	🧯 High	Implement strict credential validation, hash passwords, and enforce secure login logic
RISK-002	Data not saved in localStorage (e.g., comments, profile edits) (BUG001, BUG014, BUG028)	High – user changes lost, poor experience	Likely	🔥 High	Ensure all state changes are synced with localStorage or backend
RISK-003	Quiz does not terminate and repeats indefinitely (BUG002, BUG032)	Medium – infinite loop breaks logic, ruins UX	Likely	⚠️ High	Add end condition to quiz logic, track answered question index
RISK-004	Weak password accepted during registration (BUG017)	Critical – opens system to brute-force attacks	Possible	🧯 High	Enforce password policies (min 8 chars, symbols, numbers) client- and server-side
RISK-005	No client-side date validation allows past pickups (BUG004)	High – invalid service schedules, operational chaos	Likely	🔥 High	Validate pickup dates: enforce today + 1 minimum; disallow past

CleanCity- QA Risk Matrix Screenshot 1



CleanCity / tests / CleanCity_Risk_Matrix.md ↑ Top

Preview | Code | Blame | 27 lines (26 loc) · 7.69 KB | Raw

RISK-008	Form accessibility issues reduce usability for disabled users (BUG011, BUG045)	Medium – fails WCAG 2.1 compliance, excludes users	Possible	⚠️ Medium	Add label associations, autocomplete, proper ARIA roles
RISK-009	localStorage exposes sensitive user data in plain text (BUG018, BUG015)	Critical – serious data leak vulnerability	Likely	🔴 High	Store sensitive data securely or use session-based authentication
RISK-010	Multiple pickups can be booked for same date/time/location (BUG016, BUG012)	Medium – scheduling conflicts, resource overload	Possible	⚠️ Medium	Add backend validation for duplicates; notify users of conflict
RISK-011	LIKE/UNLIKE actions don't validate user auth (BUG027)	Medium – inconsistent tracking, inaccurate metrics	Possible	⚠️ Medium	Require login for all interactive community actions
RISK-012	Long inputs accepted where limits are defined (BUG019, BUG013)	Low – UI breaks or overflows	Possible	⚠️ Low	Set maxlength attributes and backend checks
RISK-013	Quiz reuses same questions even after restart (BUG035)	Low – poor learning outcome	Likely	⚠️ Low	Shuffle questions, track previous ones
RISK-014	Dashboard analytics not syncing real-time (BUG006)	Medium – managers see inaccurate stats	Likely	⚠️ High	Force UI re-render after successful pickup submissions

Prepared and Submitted By:

Trailblazers QA Team 3

Idah Makena

Pauline Anne Anyango

Tekra Gatwiri

Institution: PowerLearn Project Bootcamp

Date: July 16, 2025