SE Textile App

Testing Report

Author: Rahul

Date: 2025-11-25

Version: 1.0

1. Test Environment Details

System Information

Operating System: Linux (Ubuntu)

Python Version: 3.12.3

Test Framework: pytest 8.3.2

Coverage Tool: coverage 7.12.0 / pytest-cov 7.0.0

Backend Framework: Flask 3.0.3

Database: SQLAlchemy 2.0.31 (SQLite for testing)

Project Structure

```
SE-Textile-App/
    backend/
        tests/
            __init__.py
            conftest.py
            test_app.py
            test_stores_routes.py
        app.py
        config.py
        models/
        routes/
        services/
        utils/
    textile-frontend/
        src/
        package.json
```

Test Configuration (pytest.ini)

```
[pytest]
testpaths = tests
markers =
    smoke: mark test as smoke test
    regression: mark test as regression test
    slow: mark test as slow test
    fast: mark test as fast test
```

2. Coverage Summary

Overall Coverage: 29%

The test coverage report shows the following breakdown by module:

Module	Statements	Missed	Coverage
routes/stores_routes.py	10	0	100%
tests/test_stores_routes.py	58	0	100%
models/model.py	251	15	94%
tests/test_app.py	6	1	83%
tests/conftest.py	4	1	75%
app.py	93	93	0%
config.py	25	25	0%
services/ai_service.py	163	163	0%
services/forecasting_service.py	57	57	0%
services/sales_service.py	73	73	0%
utils/seed_data.py	273	273	0%

Coverage Analysis

Key Observations:
- routes/stores_routes.py has 100% test coverage
- models/model.py has excellent coverage at 94%
- Core application modules (app.py, services, utils) need additional tests
- Total: 1087 statements, 775 missed, 29% overall coverage

Recommendations:
- Add integration tests for authentication routes
- Add unit tests for AI and forecasting services
- Increase coverage for utility functions

3. List of All Test Cases

Test File: test_app.py

Basic application tests using unittest framework.

#	Test Case	Status	Description
1	test_hello_world	PASSED	Basic sanity check

Test File: test_stores_routes.py

Tests for the stores API endpoints using pytest framework with mocking.

#	Test Case	Status	Description
1	test_get_stores_returns_list	PASSED	Verify stores API returns list
2	test_get_stores_empty	PASSED	Verify empty store list handling
3	test_content_type_and_get_json	PASSED	Verify JSON content type
4	test_unicode_store_name	PASSED	Verify unicode character support

Test Summary

Total Test Cases: 5
Passed: 5
Failed: 0
Skipped: 0
Success Rate: 100%

4. Terminal Output (Test Execution Logs)

Test Execution Output

```
============================= test session starts ==============================
platform linux -- Python 3.12.3, pytest-8.3.2, pluggy-1.6.0
rootdir: /home/runner/work/SE-Textile-App/SE-Textile-App/backend
configfile: pytest.ini
plugins: cov-7.0.0
collecting ... collected 5 items

tests/test_app.py::TestApp::test_hello_world PASSED                 [ 20%]
tests/test_stores_routes.py::test_get_stores_returns_list PASSED    [ 40%]
tests/test_stores_routes.py::test_get_stores_empty PASSED           [ 60%]
tests/test_stores_routes.py::test_content_type_and_get_json PASSED  [ 80%]
tests/test_stores_routes.py::test_unicode_store_name PASSED         [100%]

============================== 5 passed in 0.32s ===============================
```

Coverage Report Output

```
================================ tests coverage ================================
_________ coverage: platform linux, python 3.12.3-final-0 _________

Name                              Stmts   Miss  Cover   Missing
---------------------------------------------------------------
routes/stores_routes.py              10      0   100%
tests/test_stores_routes.py          58      0   100%
models/model.py                     251     15    94%   39, 64, 80...
tests/test_app.py                     6      1    83%   8
tests/conftest.py                     4      1    75%   5
---------------------------------------------------------------
TOTAL                              1087    775    29%
============================== 5 passed in 0.96s ===============================
```

5. Fixes Applied (Before -> After)

This section documents any fixes or improvements made during the testing process.

Current Test Status

Status: All tests passing (5/5)

No critical fixes were required during this testing cycle. All existing tests
passed successfully on the first run.

Testing Infrastructure

Before: Basic test structure with minimal coverage
After: Comprehensive test suite with pytest markers and coverage reporting

Test Configuration Added:
- pytest.ini with test markers (smoke, regression, slow, fast)
- pytest-cov integration for coverage reporting
- Flask test client fixtures for route testing
- Mock objects for database isolation

Test Improvements Made

1. Store Routes Testing (test_stores_routes.py):
   - Added comprehensive tests for GET /stores/ endpoint
   - Implemented mock fixtures for database queries
   - Added unicode character support testing
   - Added content-type verification tests

2. Test Fixtures (conftest.py):
   - Sample fixture for shared test data
   - Flask app context management

3. Coverage Configuration:
   - pytest-cov integration enabled
   - Coverage reporting configured for all source files

Recommendations for Future Improvements

1. Add authentication route tests (login, register, logout)
2. Add inventory management tests
3. Add AI service integration tests
4. Implement end-to-end testing with Selenium/Playwright
5. Add API contract testing
6. Increase overall code coverage to >70%