Enhancing Sugar Labs' Testing Framework for Music Blocks

Basic Details

Name: Nnadi Ugochukwu
Email: ugonnnadi45@gmail.com
Location: California, United States
Time Zone: Pacific Daylight Time (UTC-7)
Language: English

About Me

I am an experienced Software Testing and Quality Assurance Specialist with a strong background in frontend testing and CI/CD automation. Over the years, I have designed and executed test plans that ensure high software reliability, integrated automated testing into continuous deployment pipelines, and enhanced quality assurance processes across various open-source projects. My expertise includes React Testing Library, Jest, and GitHub Actions, which are key to testing frontend-heavy applications like Sugar Labs' Music Blocks. I am passionate about empowering developers to ship robust, user-friendly applications and mentoring contributors to adopt reliable testing strategies.

Executive Summary

This project proposes an enhancement of the testing infrastructure for Sugar Labs' Music Blocks v4. The application is a browser-based, frontend-only visual programming language built with TypeScript, JavaScript, and React. My primary focus will be improving unit and component composition testing using Jest and React Testing Library, along with automating these tests within a CI/CD pipeline. By the end of this project, the Music Blocks codebase will have higher test coverage, more reliable UI logic, and an improved contributor experience through clear testing documentation.

Benefits to the Community

- **Higher Code Reliability:** Comprehensive tests catch bugs earlier in the development cycle.
- **Faster Development Cycles:** CI-integrated testing speeds up reviews and deployments.

- **Improved Contributor Experience:** Clear test documentation and automation lower the entry barrier for new developers.

Deliverables

- **Expanded Unit Tests** using Jest for core React components and utility logic.
- **Component Composition Tests** for interactive UI flows using React Testing Library.
- **CI/CD Integration** with GitHub Actions for automated testing.
- **Test Coverage Monitoring** with coverage thresholds enforced.
- **Testing Documentation** to guide new and existing contributors.

Background

Music Blocks is designed to help learners understand programming through music. The transition to `musicblocks-v4` introduces modern frameworks and modular architecture, emphasizing the need for comprehensive frontend testing. Presently, the project lacks sufficient unit and composition testing coverage. This proposal aims to fill this gap by applying robust testing strategies and automation practices to solidify the codebase for future development.

Design and Description of Work

1. Unit Testing with Jest

Jest will be used to validate individual component logic, especially for reusable components and logic-heavy utilities.

```
import { render, screen } from '@testing-library/react';
import PlayButton from '../components/PlayButton';

test('renders play button correctly', () => {
  render(<PlayButton />);
  expect(screen.getByRole('button', { name: /play/i })).toBeInTheDocument();
});
```

2. Component Composition Testing

React Testing Library will help test component interactions and layout rendering.

```
import { render, fireEvent, screen } from '@testing-library/react';
import MusicEditor from '../components/MusicEditor';

test('adds a new note to the editor', () => {
  render(<MusicEditor />);
  fireEvent.click(screen.getByText('Add Note'));
  expect(screen.getByText('Note 1')).toBeInTheDocument();
});
```

3. CI/CD Integration

Tests will be automated with GitHub Actions to ensure reliability on every commit or PR.

```
name: Run Tests
on: [push, pull_request]
jobs:
  test:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Install Dependencies
        run: npm install
      - name: Run Jest Tests
        run: npm test
```

Timeline

Period (Weeks)	Dates	Task
Week 1-2	June 2 – June 15, 2025	Review current test coverage. Identify components lacking tests.
Week 3-4	June 16 – June 29, 2025	Write unit tests for core components using Jest.

Week 5-6	June 30 – July 13, 2025	Add component composition tests using React Testing Library.
Week 7-8	July 14 – July 27, 2025	Integrate CI with GitHub Actions. Set coverage thresholds.
Week 9-10	July 28 – August 10, 2025	Optimize test performance and resolve flaky tests.
Week 11-12	August 11 – August 25, 2025	Finalize documentation. Conduct contributor onboarding session.

Expected Outcomes

- Full unit and UI composition test coverage for key user-facing components.
- Stable automated test execution integrated into CI/CD.
- Contributor guidelines and test writing docs published.
- Community contributors better equipped to maintain code quality.

Future Work

- Include visual regression testing.
- Extend test coverage to storybook and accessibility tools.
- Explore integration with educational testing tools.

How many hours will you spend each week on your project?

If selected, I will be working on GSoC full-time. I am able to put in 30 hours per

week on this project, and I am open to putting in more hours if the project demands it.

How will you report progress between evaluations?

I will sync with the project mentors regularly, via teleconferencing apps such as zoom, google meet and via the communities channel, as well.

Discuss your post GSoC plans. Will you continue contributing to Sugar Labs after GSOC ends ?

yes, I plan to remain an active contributor to Sugar Labs, as there are still areas for improvement, as i listed in the future work section above

Why Me?

- Proven experience in frontend QA and automated testing for React applications.
- Skilled in setting up CI/CD pipelines that enforce code quality.
- Strong focus on building maintainable, contributor-friendly open-source projects.
- Passionate about accessible learning tools and user-first engineering.

By focusing on Music Blocks' critical frontend architecture, this project ensures a robust user experience for educators and learners alike while empowering contributors with modern testing tools and workflows.