**Quality Assurance Plan**

The **Quality Assurance Plan (QAP)** outlines the strategies, activities, and tools used to ensure the Book Exchange System meets the required quality standards.

**1. Objectives**

The goal is to deliver a reliable, functional, and user-friendly system by:

* Ensuring core features like book catalog management, reservations, and user interactions work correctly.
* Identifying and fixing bugs early in the development process.
* Validating performance and usability for all users.

**2. Scope**

The QAP will focus on:

* Backend components (Book, User, Catalog) to ensure their functions (e.g., adding books, reserving them) operate smoothly.
* Integration between backend and frontend (e.g., Flask communicating with HTML forms).
* UI testing to verify the interface displays book details and handles user inputs correctly.

**3. Testing Methodology**

* **Unit Testing:** Verify each module (e.g., Book.reserve(), Catalog.add\_book()) independently using pytest.
* **Integration Testing:** Test the flow between modules and components, such as users reserving books and their status updating.
* **User Interface Testing:** Use Selenium to simulate and test user actions (e.g., searching books, reserving them).
* **Regression Testing:** Ensure new changes do not break existing functionality.
* **Performance Testing:** Validate the system's responsiveness under typical workloads.

**4. Tools and Metrics**

* **Tools:**
  + pytest for backend tests.
  + Selenium for frontend and UI testing.
  + GitHub Actions for automated testing workflows.
* **Metrics:**
  + Test Coverage: Ensure at least 80% of the code is tested.
  + Bug Counts: Track and resolve issues identified during testing.
  + Defect Density: Measure the number of defects per module.
  + Test Execution Time: Assess how quickly tests complete.

**Sprint and Task Planning**

**Sprint and Task Planning** ensures the project is broken into smaller, manageable parts with clear goals, deadlines, and responsibilities.

**1. Sprint Overview**

Development is divided into **4 sprints**, each focusing on a specific goal:

* **Sprint 1:** Implement core backend functionality (book catalog and reservation system).
* **Sprint 2:** Add user interaction features and integrate the backend with a simple UI.
* **Sprint 3:** Enhance the frontend (HTML/Flask integration) and build advanced features like search.
* **Sprint 4:** Perform rigorous testing (unit, integration, and UI), fix bugs, and prepare for final deployment.

**2. Tasks by Sprint**

Below are the tasks for each sprint:

1. **Sprint 1: Backend Development**
   * Implement the Book, User, and Catalog classes.
   * Add methods for reserving, releasing, and managing books.
   * Write unit tests for backend features.
2. **Sprint 2: User Interaction Features**
   * Integrate Flask with backend modules.
   * Build a basic HTML interface for displaying books and their status.
   * Implement forms for searching and reserving books.
3. **Sprint 3: Frontend Integration**
   * Design and enhance the HTML interface for better usability.
   * Add interactive features (e.g., live status updates) using JavaScript.
   * Test the interaction between the UI and backend with Flask.
4. **Sprint 4: Testing and Deployment**
   * Perform unit, integration, and UI testing.
   * Fix bugs and improve the system based on testing results.
   * Deploy the application on a local or cloud-based server.

**3. Tools for Task Management**

* **Trello, Jira, or GitHub Projects:** Track sprint progress, assign tasks, and set deadlines.
* **Version Control (Git):** Manage code changes and collaborate with team members effectively.
* **Daily Standups:** Review progress, address blockers, and adjust plans.