### **Testing and Quality Assurance**

**Are there specific testing methodologies or tools you want included in the master document?**

While your existing documents mention Sentry for error tracking and touch upon system resilience, for a comprehensive testing and quality assurance strategy to be included in your master document, I would recommend the following methodologies and tools:

* **Unit Testing:**
  + **Methodology:** Focus on testing individual components or functions in isolation to ensure they work as expected. This helps catch bugs early in the development cycle.
  + **Tools:**
    - **Frontend (React Native):**
      * **Jest:** A JavaScript testing framework, widely used for React Native. Excellent for component testing, snapshot testing, and basic logic testing.
      * **React Native Testing Library:** Builds on Jest, focusing on testing user-facing behavior rather than implementation details, encouraging more robust tests.
    - **Backend (FastAPI):**
      * **Pytest:** A powerful Python testing framework. It's highly extensible and makes writing simple, readable tests easy. It integrates well with FastAPI.
      * **httpx (for FastAPI):** For making asynchronous HTTP requests to test FastAPI endpoints programmatically.
* **Integration Testing:**
  + **Methodology:** Verify that different modules or services in your application interact correctly. This includes testing API endpoints, database interactions, and integrations with external services (like AI APIs or Redis).
  + **Tools:**
    - **Backend (FastAPI with Pytest):** Use Pytest to write tests that span across multiple components, e.g., testing if an API call correctly updates the PostgreSQL database or interacts with Redis. Mock external services (like AI APIs) where appropriate to ensure tests are fast and reliable.
    - **Docker Compose:** For setting up a local testing environment that mirrors your production stack (including PostgreSQL, Redis, and FastAPI services) to run integration tests against.
* **End-to-End (E2E) Testing:**
  + **Methodology:** Simulate real user scenarios across the entire application stack, from the frontend UI to the backend services and database. This confirms the complete user flow works as intended.
  + **Tools:**
    - **Appium:** An open-source test automation framework for mobile applications (iOS, Android, and hybrid apps). It allows you to write tests in various languages and simulate user interactions on real devices or emulators.
    - **Detox (for React Native):** A gray box end-to-end testing and automation framework for React Native. It's known for its speed and reliability for mobile E2E tests.
* **Performance Testing:**
  + **Methodology:** Evaluate the application's responsiveness, stability, and scalability under various load conditions. This is crucial for AI-driven features and asynchronous tasks.
  + **Tools:**
    - **Locust:** An open-source load testing tool. You define user behavior in Python code, and then Locust simulates millions of concurrent users. Useful for testing API endpoints and identifying bottlenecks.
    - **JMeter:** A widely used open-source tool for load and performance testing.
* **Security Testing:**
  + **Methodology:** Identify vulnerabilities in the application that could be exploited by attackers. This includes penetration testing, vulnerability scanning, and code reviews.
  + **Tools:**
    - **OWASP ZAP:** An open-source web application security scanner.
    - **Snyk/Bandit (for Python):** Tools for identifying security vulnerabilities in your codebase and dependencies.
* **Manual/Exploratory Testing:**
  + **Methodology:** Human testers interact with the application to discover bugs and usability issues that automated tests might miss. Especially important for the subjective experience of AI interactions and philosophical tones.
  + **Tools:** None specific, but clear bug reporting tools (e.g., Jira, GitHub Issues) are essential.

Incorporating these methodologies and tools will provide a comprehensive testing framework, ensuring high quality and stability for your Setarcos app.