Financial Playbook Application - Final Development Plan (Single-User Focus)

Project Name: Financial Playbook
Client: Superior Networks LLC (Dwain Henderson Jr.)
Deployment Target: Vercel
Document Version: 2.0 (Single-User)
Date: November 23, 2025

Executive Summary

This document outlines the revised, comprehensive development plan for the Financial Playbook application, now streamlined for a **single-user architecture**. This strategic shift significantly reduces complexity, accelerates initial delivery, and allows for a focused effort on the core financial engine and AI capabilities. Multi-user functionality, including delegated access, will be deferred to a future development phase.

A new critical requirement has been added: **all AI-generated mathematical calculations will be rigorously validated** using traditional, deterministic programming methods to ensure 100% accuracy and reliability.

Key Changes from Multi-User Plan

1. **Single-User Architecture:** The entire application will be built for a single user (Dwain Henderson Jr.). This removes the need for row-level security, complex permission systems, user management tables, and data isolation logic, simplifying the database schema and backend code.
2. **Deferred Features:** The following features are **postponed** to a future multi-user phase:
    - Delegated Access for Tax Pros/External Users

    - AI Delegation System (delegating tasks to other users)
    - Delegate Profiles and granular permissions
3. **AI Math Validation:** A new, mandatory step is added to all AI-related workflows. After an AI model extracts or calculates a number (e.g., a total from a bank statement), a separate, non-AI function will perform the same calculation to verify the result. Any discrepancy will be flagged for user review.

Security Analysis (Single-User Context)

Security remains paramount. The focus shifts from multi-user authorization to protecting a single user’ s data from external threats.

- **Authentication:** Google OAuth remains the secure entry point. Session management will be robust.
- **Data Security:** All previous mitigations (encryption, XSS prevention, SQL injection prevention) are still critical.
- **API Security:** All API keys will be stored securely as environment variables.
- **File Upload Security:** All file uploads will be scanned and stored in isolated storage.

The removal of multi-user features simplifies the attack surface, making it easier to secure the application effectively.

Revised Phased Development Plan

Phase 1: Project Setup & Basic Authentication (1 week)

Objective: Establish a simplified project structure and a secure authentication system for a single user.

Tasks:

1. Initialize Next.js project.

2. Set up PostgreSQL database and Prisma ORM with a simplified schema (no multi-user tables).
3. Implement Google OAuth 2.0 using NextAuth.js.
4. Build basic security headers and session management.

Deliverable: A secure, functional login system for a single user.

Phase 2: QuickBooks Integration & Financial Data Engine (2 weeks)

Objective: Integrate with QuickBooks Online and build the core financial data engine.

Tasks:

1. Implement QuickBooks OAuth 2.0 connection.
2. Build the data sync service to fetch accounts, transactions, and bills.
3. Implement webhook receiver for real-time updates.
4. Build the core UI for viewing accounts and transactions.
5. Implement sync controls (manual, on-login, scheduled).

Deliverable: A dashboard displaying real-time financial data from QuickBooks.

Phase 3: AI Strategy Builder with Math Validation (2-3 weeks)

Objective: Build the AI-powered document analysis tool with a mandatory math validation layer.

Tasks:

1. Build the file upload interface.
2. Implement the AI document processing pipeline (OCR, data extraction).
3. **Crucially, build a separate validation service.** After the AI extracts transactions, this service will programmatically re-calculate totals and balances.
4. If validation fails, the UI will flag the discrepancy and ask the user to confirm the numbers.
5. Implement the AI-powered strategy recommendation engine.

Deliverable: A functional Strategy Builder Wizard that can ingest documents and produce validated, AI-driven strategic recommendations.

Phase 4: Logic Simulator & Validated Report Generator (2-3 weeks)

Objective: Build the simulation engine and the report generator, ensuring all report calculations are validated.

Tasks:

1. Build the Scenario Logic Editor UI.
2. Implement the day-by-day simulation engine.
3. Build the Report Generator.
4. **Implement AI Math Validation:** After the AI generates analysis (e.g., a projected balance), the report generator will use the raw simulation data to perform its own calculation. The final report will only show the validated numbers.
5. Implement report export functionality.

Deliverable: The ability to run complex simulations and generate accurate, validated financial reports.

Phase 5: Automation, Audit Trail & Final UI (1-2 weeks)

Objective: Implement simplified automation, a single-user audit trail, and the final UI with QC numbering.

Tasks:

1. Implement the simplified automation system (for recurring bills, without delegation).
2. Implement the price variance alerts.
3. Build the audit trail, logging all significant actions by the primary user.
4. Complete all UI pages, implementing the QC numbering system for easy reference.
5. Implement the development mode toggle to hide QC numbers.

Deliverable: A feature-complete UI with automation and a comprehensive audit log.

Phase 6: Security Review, Testing & Deployment (1 week)

Objective: Conduct a final security audit, perform testing, and deploy the application to Vercel.

Tasks:

1. Conduct a final security audit focused on single-user vulnerabilities.
2. Perform comprehensive end-to-end testing of all features.
3. Set up the production environment on Vercel.
4. Deploy the application.
5. Create user documentation and conduct a final handover.

Deliverable: A secure, production-ready, single-user Financial Playbook application deployed on Vercel.

Conclusion & Next Steps

This revised single-user plan dramatically reduces initial complexity and risk, allowing for a faster path to a functional and valuable product. It prioritizes the core engine and the accuracy of its outputs. The new estimated timeline for this streamlined build is **7-10 weeks**.

This plan represents the definitive roadmap for development. Upon your approval, I will proceed with Phase 1 and begin building the application.