MVP Release Roadmap
Small-Business Financial Planning Software

This roadmap expands the build sequence into a more complete path to a credible MVP release. The key idea is to avoid letting downstream features, admin tools, pricing, legal work, and beta testing pull the project away from the core product proof: can a small business owner use the application to produce a reasonable financial plan and final outputs that reconcile end to end?

1. Executive Recommendation

Your current sequence is directionally right: prove Apps 1 through 4 first, then build the transaction capture and trial balance layer, then generate the financial statements and final report outputs. The main improvement I would make is to add a formal readiness layer before development continues too far. The MVP should be governed by toll gates, case-study evidence, legal and payment readiness, data controls, support processes, and beta criteria.

Priority	Recommendation	Reason
1	Freeze the MVP scope around the financial planning path, not the full product vision.	This prevents AI agents, developers, and stakeholders from expanding the build before the core workflow is proven.
2	Make Apps 1 through 4 and App 5 the accounting engine proof point.	If source assumptions, transaction generation, and trial balance do not reconcile, later statements and reports will inherit errors.
3	Define pass/fail test evidence for every toll gate.	The project should not move forward based on visual completion or partial functionality.
4	Separate beta readiness from public release readiness.	A beta can tolerate some operational friction; a paid release needs support, compliance, onboarding, payments, and admin controls.
5	Build the admin panel earlier than the final public release, but not before the core financial engine is proven.	You need enough admin visibility for beta support, but the admin panel should not consume resources before the accounting workflow is stable.

2. Suggested MVP Release Phases

The following sequence keeps your existing build logic but adds the missing control points that normally determine whether an MVP is actually ready for beta users and, later, a paid release.

Phase	Workstream	What Must Happen	Exit Criteria
Phase 0	MVP Scope Lock	Define MVP promise, user personas, included apps, excluded features, accounting basis, assumptions, and success criteria.	Approved MVP scope document and change-control rule.
Phase 1	Build 1: Apps 1–4	Complete all function points and transaction generation for the foundation apps. Do not spend major resources on downstream apps yet.	Apps 1–4 operate end to end for both product and service case studies.
Phase 2	Functional Testing: Apps 1–4	Run Product Company and Service Company Approach 2 through all key use cases and transaction outputs.	All expected calculations, user inputs, and generated transactions pass.
Phase 3	Toll Gate 1	Confirm Apps 1–4 have passed functional testing and	Written pass/fail evidence and issue log signed off

		transaction processing for both case studies.	before Build 2.
Phase 4	Build 2: App 5 Transaction Capture + Trial Balance	Design transaction capture model, modified cash basis framework, accrual entries, and closing entry process.	Trial balance framework accepts transactions from Apps 1–4 and supports adjustments.
Phase 5	Functional Testing: App 5	Extend both case studies into App 5 and compare trial balance outputs against expected financial statement logic.	Two trial balances reconcile before statement builds begin.
Phase 6	Toll Gate 2	Lock the accounting engine for MVP purposes.	Transaction capture, adjustments, and trial balance pass both case studies.
Phase 7	Build 3: App 6 Monthly Income Statement	Develop monthly income statement and link it to the adjusted trial balance.	Income statement matches expected case-study outputs.
Phase 8	Build 4: App 7 Balance Sheet	Develop fiscal year-end balance sheet with correct treatment of accrual entries and no-receivables footnote if appropriate.	Balance sheet balances and agrees to adjusted trial balance.
Phase 9	Build 5: App 8 Direct Monthly Cash Flow	Develop direct monthly cash flow statement and cash optimization routines.	Cash flow detects insufficient cash, excessive ending cash, and recommends practical adjustments.
Phase 10	Financial Statement Integration Testing	Test income statement, balance sheet, and cash flow together as one reporting package.	Statements reconcile, cash connects, and case-study outputs are explainable.
Phase 11	Build 6: PDF Report + Excel Model	Create final client-facing PDF report and downloadable Excel model.	Outputs are professional, consistent, and match app data.
Phase 12	End-to-End Testing	Run both case studies from first input through final PDF and Excel outputs.	No blocking defects; all outputs agree with expected results.
Phase 13	Legal, Compliance, and Risk Review	Finalize privacy policy, terms, consent language, disclaimers, data handling, and beta participation terms.	Legal content ready for beta users before external testing.
Phase 14	Pricing and Packaging Test	Decide what beta should test: pricing, base-year model, three-year model, paid tiers, or free trial assumptions.	Pricing assumptions documented, even if not fully finalized.
Phase 15	Build 7: Hub Page	Finalize the user hub so beta users know where to start, what is complete, and what actions remain.	Hub page guides users through the MVP workflow without developer help.
Phase 16	Build 8: Admin Control Panel for Beta	Develop the minimum admin features needed to monitor users, support issues, outputs, subscriptions, and feedback.	Admin can see beta users, status, issues, and basic product usage.
Phase 17	Beta Readiness Toll Gate	Confirm functionality, legal, onboarding, support, payment assumptions, admin visibility, and test scripts are ready.	Approved beta launch checklist.
Phase 18	Controlled Beta	Run a limited beta with selected small business owners or advisors using defined test objectives.	Feedback, defects, support issues, and product confusion are captured.
Phase 19	Beta Fix Cycle	Prioritize defects and improvements into must-fix, should-fix, later, and not-in-	Only release-blocking items are fixed before MVP release.

		MVP categories.	
Phase 20	MVP Release Readiness	Confirm product, reporting, support, onboarding, admin, compliance, pricing, and payment readiness.	Go/no-go decision for limited paid MVP release.

3. Missing Steps I Would Add

The current list is strong on accounting-function sequencing, but it needs more operational, product, and release controls. These are the items I would add before calling the software a strong MVP.

Missing Area	Add This Step	Why It Matters
MVP scope control	Create a formal MVP scope lock and backlog parking lot.	Prevents downstream apps and nice-to-have features from weakening the core release.
User workflow	Map the full user journey from signup through final report download.	A technically correct app can still fail if users do not know what to do next.
Assumption governance	Create an assumptions table for each case study and each app.	Financial plans depend on assumptions; users and support need to know what drives results.
Data model	Document the source-of-truth tables and transaction flow across Apps 1–8.	This helps prevent reconciliation problems and duplicate logic.
Calculation controls	Build a calculation verification checklist for each app.	This makes testing repeatable instead of dependent on memory.
Error handling	Define user-facing error messages for bad inputs, missing data, and impossible financial outputs.	Small business users need guidance, not technical failures.
Onboarding	Create a short guided onboarding path and sample case-study walkthrough.	Beta users will need help understanding what the software is asking for.
Security and privacy	Confirm login, role access, data storage, privacy policy, consent, and deletion process.	Financial planning data is sensitive even when it is not full bookkeeping data.
Payments	Decide whether Stripe is active in beta or simulated until paid launch.	Payment complexity can distract from product validation if introduced too early.
Support process	Define how beta users submit problems and how issues are triaged.	Without support workflow, beta feedback becomes scattered and hard to act on.
Admin visibility	Build minimum admin reporting before beta or immediately after first beta users.	You need to see where users get stuck and whether outputs complete.
Release governance	Create go/no-go criteria for beta and MVP release.	This avoids emotional decisions based on excitement instead of readiness.

4. Toll Gates and Pass/Fail Criteria

The most important improvement is to make every toll gate evidence-based. A toll gate should not mean that the screen appears to work. It should mean that a specific case study, calculation, transaction set, and output package has passed a repeatable test.

Toll Gate	Decision Question	Minimum Evidence Required	Decision
TG1: Apps 1–4	Can the foundation apps generate correct transactions for both case studies?	Completed test script, expected vs. actual results, defect log, and no open critical defects.	Proceed to App 5 or return to Build 1.
TG2: App 5	Does the transaction capture model and trial balance framework reconcile?	Two completed trial balances with adjustments and closing entries tested.	Proceed to financial statements or fix accounting engine.
TG3: Statements	Do income statement, balance sheet, and cash flow	Reconciled reporting package for both case	Proceed to PDF/Excel outputs or fix reporting logic.

	agree with the trial balance and each other?	studies.	
TG4: Final Outputs	Do PDF and Excel outputs match the application data and present a clear financial plan?	Final report package generated from both case studies and reviewed for clarity.	Proceed to end-to-end testing or fix outputs.
TG5: Beta Readiness	Can selected users test the product safely and productively?	Legal language, onboarding, support, admin visibility, known-issues list, and feedback process complete.	Launch controlled beta or delay.
TG6: MVP Release	Is the product ready for a limited paid or public MVP release?	Resolved beta blockers, pricing decision, payment readiness, support process, and release checklist complete.	Release or run another beta cycle.

5. Recommended Beta Test Objectives

Your beta should not be limited to whether buttons work. It should answer whether a real small business owner can understand the flow, enter reasonable assumptions, trust the output, and see enough value to consider paying for the product.

Beta Objective	What to Test	Success Indicator
Functional accuracy	Apps 1–8, final PDF, and Excel output.	Users can complete the workflow without blocking defects.
Financial reasonableness	Whether the plan outputs make sense to a business owner or advisor.	Outputs are explainable and do not create obvious accounting concerns.
UX clarity	Navigation, hub page, instructions, input screens, and error messages.	Users know what to do next without live coaching.
Product packaging	Base-year model, three-year model, and what should be included in MVP.	Users can explain which version they would value.
Pricing	Trial, one-time price, subscription, advisor model, or tiered pricing.	Beta users provide credible feedback on willingness to pay.
Support burden	How often users need help and what kind of help they need.	Support issues are manageable and repeatable fixes are identified.
Admin needs	What you need to see as owner/operator.	Admin panel supports beta monitoring, troubleshooting, and user status review.

6. Practical Build Order Recommendation

If I were simplifying the roadmap into a working sequence, I would group the project into five major release blocks. This is easier to manage than treating every app as an isolated build.

Release Block	Included Builds	Primary Goal
Block A: Foundation Engine	Apps 1–4 plus functional testing	Prove that inputs and transaction generation work for the two case studies.
Block B: Accounting Core	App 5 transaction capture, trial balance, accruals, and closing entry	Prove that the accounting engine can reconcile.
Block C: Financial Statements	Apps 6–8	Produce monthly income statement, balance sheet, and direct cash flow that agree with the trial balance.
Block D: Final User Outputs	PDF report, Excel model, hub page, onboarding, and error handling	Give the user a complete and understandable financial plan package.
Block E: Beta and Operating Layer	Legal, beta scope, admin panel, support process, Stripe/payment decision, and release checklist	Prepare the product to operate with real users.

7. My Suggested Revised Numbered List

Below is a cleaned-up version of your list with several missing items added. This can become the master build roadmap for the MVP.

1. Define MVP scope, target user, case-study assumptions, accounting basis, and features that are excluded from MVP.

2. Create master architecture map showing Apps 1–8, source data, transaction flow, trial balance, statements, PDF, Excel, hub page, admin panel, and Stripe/payment touchpoints.

3. Build 1: Complete Apps 1–4 with all function points and transaction generation.

4. Functional test Apps 1–4 using Product Company and Service Company Approach 2 case studies.

5. Toll Gate 1: Apps 1–4 pass all function-point and transaction-processing tests for both case studies.

6. Build 2: Complete App 5 transaction capture model, modified cash basis transaction framework, accrual entries, and closing entry process.

7. Functional test App 5 by extending both case studies into trial balance outputs.

8. Toll Gate 2: Both trial balances reconcile and agree with expected case-study financial results.

9. Build 3: Complete App 6 monthly income statement and reconcile it to the adjusted trial balance.

10. Build 4: Complete App 7 fiscal year-end balance sheet and confirm balance sheet agrees with the adjusted trial balance.

11. Build 5: Complete App 8 direct monthly cash flow statement and optimization routines for insufficient or excessive cash balances.

12. Financial statement integration test: income statement, balance sheet, and cash flow reconcile as one reporting package.

13. Toll Gate 3: Full financial statement package passes both case studies.

14. Build 6: Create PDF report and Excel model outputs.

15. Output testing: confirm PDF and Excel outputs match application data, financial statements, and case-study expectations.

16. Build user onboarding, help text, error handling, and sample walkthrough for small business owners.

17. Finalize legal and regulatory readiness for beta: privacy policy, terms, user consent, disclaimers, data handling, and beta agreement.

18. Define beta objectives and scope, including functionality, pricing, base-year versus three-year model, UX/UI, hub page, and final output usefulness.

19. Build 7: Finalize hub page before beta testing so users can navigate the workflow clearly.

20. Build 8: Develop minimum admin control panel for beta monitoring, user status, support issues, output completion, and early usage visibility.

21. Decide Stripe/payment approach for beta: active payments, test mode, manual invoicing, free beta, or waitlist-only pricing test.

22. Run full end-to-end internal testing from first input through PDF and Excel output for both case studies.

23. Toll Gate 4: Beta readiness approval based on product, legal, support, admin, onboarding, and known-issues checklist.

24. Conduct controlled beta with a limited user group and structured feedback process.

25. Triage beta feedback into must-fix, should-fix, later, and not-in-MVP categories.

26. Complete beta fix cycle and repeat end-to-end regression testing.

27. Toll Gate 5: MVP release readiness decision.

28. Launch limited MVP release with monitoring, support workflow, release notes, and issue tracking.

8. Immediate Next Actions

The next best move is not to build more screens. It is to turn the roadmap into a controlled test-and-release plan. I would start with the following actions.

Next Action	Owner Question	Output
Create MVP scope lock	What exactly must the first MVP do, and what is out of scope?	One-page MVP scope document.
Create case-study test workbook	What are the expected results for the Product Company and Service Company?	Expected vs. actual testing file.
Create architecture map	Where does each transaction originate, transform, and report?	End-to-end architecture diagram.
Create toll-gate checklist	What evidence is required before moving to the next build?	Pass/fail checklist for each toll gate.
Define beta operating model	Who will test, what will they test, and how will issues be captured?	Beta plan and user feedback process.

9. Bottom Line

The MVP should be judged by one central standard: can a small business owner move through the workflow and receive a clear, reasonable, and internally consistent financial plan? If the answer is yes for the two case studies, then you have a product foundation. If the answer is no, then downstream features, pricing, and admin tools will not fix the problem. Build the accounting engine first, prove it with case studies, then prepare the user experience, legal framework, admin layer, and beta process around that core.