How I scope an MVP in one call: my question list

Muhammad Gulzaib
https://orcid.org/0009-0000-8709-6989
SynergyFlow Labs · Islamabad, Pakistan

Published	2026-08-20
URL	https://gulzaib.vercel.app/blog/how-i-scope-an-mvp-in-one-call-my-question-list
Licence	© 2026 Muhammad Gulzaib. All rights reserved. Code samples MIT.

ABSTRACT

A practical first-call checklist for scoping an MVP, from the user's problem and core workflow to risks, estimates and a one-page proposal.

Keywords: saas, product, validation, lessons

> **TL;DR**
>
> A first call can produce a useful MVP outline if it ends with one user, one problem and one complete workflow. Ask about the current workaround, the first release, risks and budget, then send a written scope with open questions. Treat the result as a starting point, not a validated specification or a fixed-price promise.

This guide was prepared in November 2026. The displayed 20 August 2026 date was requested for the post; it is not its original publication date.

A client can arrive with a list of screens and still be unsure which problem the product must solve first. The job of a scoping call is to turn that list into a decision: what should the first user be able to finish, and what can wait?

The questions below are a proposed checklist, not a transcript of a client call. The school example is fictional. Download the call checklist to use it with your own notes.

What one call can and cannot do

An MVP, or minimum viable product, is a small usable version built to test an important assumption with real users. The call can identify that assumption and sketch a first release. It cannot prove that customers want the product or reveal every integration risk.

The Government Digital Service recommends understanding users and constraints before committing to a solution. Its discovery guidance is aimed at government services and describes a broader process than one call, but the distinction is useful for a small product

too. [1]

I would use this 60-minute agenda as a starting point, adjusting it when the problem is more complicated:

Time	Topic	Leave with
0–10 minutes	Problem and user	One person and the problem they face
10–25 minutes	Current workflow	What happens today and where it breaks
25–40 minutes	First release	One end-to-end task and exclusions
40–50 minutes	Constraints and risk	Data, security, integrations and unknowns
50–60 minutes	Budget and next steps	Decision-maker, constraints and follow-up

If the person doing the work is not on the call, record that as a gap and arrange to talk to them. The buyer's explanation is useful, but it is not a substitute for observing the user.

1. Find the problem and the first user

Start with the situation, not the requested feature list:

1. Who will use this first? What is their role?
2. What are they trying to finish, and when does the problem occur?
3. What happened the last time they tried to do it?
4. What happens if nothing changes for another three months?
5. Who buys or approves the product, and who uses it daily?

"Schools need a dashboard" is a solution idea. "The fee clerk needs to prepare accurate monthly vouchers without checking three registers" is a problem you can investigate. Ask for an example with personal information removed, rather than assuming that a principal and a clerk describe the same problem.

GOV.UK's user research guidance distinguishes the user's need from a proposed solution and recommends interviewing or observing actual users. A first call supplies hypotheses for that research. [2]

2. Walk through the current work

Ask the client to take you through one recent task from start to finish:

1. What starts the task and what information is needed?

2. Which register, spreadsheet or app is used now?
3. Where does someone wait, copy information or check it by hand?
4. How often does the task happen, and what does a mistake cost?
5. What happens when the internet or power is unavailable?

Write down the steps in the client's words. If the answer is "we do everything on WhatsApp", ask who sends the message, what happens next and where the final record is kept. Ask permission before seeing real data; anonymised examples are usually enough for the first discussion.

The outcome might be a printed challan, a confirmed booking or a receipt. A workflow is not finished merely because the user can log in. It is finished when the person has the result they came for.

3. Choose the smallest complete release

Now turn the problem into a first workflow:

1. What must the first user do from start to finish?
2. What result tells them the task is complete?
3. What can the team do manually for a small pilot?
4. Which requests can wait without blocking that task?
5. What would make this release unsafe or unusable?

Consider a fictional school that wants software for fees, attendance and results. Its first test could be whether one office can prepare and print accurate monthly fee vouchers. That is narrower than an entire school ERP, but it still includes the steps needed to produce a useful voucher.

For the fictional pilot	Decision
First user	Fee clerk at one school
Core workflow	Enter student and fee details, apply an agreed concession, review and print a voucher
In scope	Staff sign-in, restricted access, fee entry, voucher review and printing, backup and recovery plan
Outside this pilot	Attendance, result cards, parent app and automatic bank reconciliation
Manual for now	School checks a small sample of balances before printing; vendor assists with initial import
Open question	Can the office share an anonymised voucher and fee rules?

A manual step needs an owner and a limit. If the vendor assists with import, say which file format and how many records the pilot covers. Manual processing is not an excuse to handle student data casually or quietly promise unlimited support.

4. Ask about the risks before discussing a deadline

A short feature list can hide long dependencies. Ask these before putting a price in writing:

1. What data will the product hold, and who may see or change it?
2. Do we need payments, messaging or another system for the first workflow?
3. Who owns those accounts, and can we test access before committing?
4. What must work on a phone, in Urdu or during a connection failure?
5. What happens if someone makes a mistake or data needs restoring?

Separate a real requirement from an assumption. "It must work offline" needs a follow-up: which actions must still work, on which devices, and how will changes be reconciled later? A read-only copy, a printable backup and full offline editing are different scopes.

Security, privacy and recovery belong in the first release when real people depend on it. Agree access controls and data handling appropriate to the information involved. OWASP's Application Security Verification Standard provides a starting point for specifying and testing web application security requirements; it is not a substitute for assessing the project's own risks. [4]

If an integration is essential, check its documentation, access process and costs. If access is not available yet, write it down as a dependency. Do not treat an untested integration as a fixed number of development days.

5. Discuss budget, decisions and the next step

Ask the commercial questions directly:

1. What budget range has been set aside for the first release?
2. Is there a real deadline, and what happens if it moves?
3. Who approves the scope and who signs off the work?
4. Who supplies content, records, accounts and feedback?
5. How will hosting, third-party charges and ongoing support be paid for?

A first-call estimate is a range with assumptions, not a guarantee. If the workflow is clear but the payment integration is unknown, estimate the understood work separately and propose a short investigation for the unknown. Once access and requirements are checked, revise the estimate before either side commits.

For example, an illustrative estimate might be 40–60 hours for a defined pilot, plus a

separately scoped integration. At a hypothetical PKR 2,000 per hour, that understood portion would be PKR 80,000–120,000 before third-party charges and tax if applicable. These figures are only arithmetic examples, not my rates or a quote. Agree what happens if the work changes and whether the client is buying a fixed scope or paying for time.

Turn the answers into a one-page scope

Send the summary after the call and ask the decision-maker to correct it:

- **User and problem:** Who is affected, and what are they trying to do?
- **First workflow:** Where does it start, and what result completes it?
- **Included and excluded:** What will be delivered, and what is explicitly deferred?
- **Acceptance:** What must the client be able to check before sign-off?
- **Constraints and dependencies:** Devices, access, data, security and third parties.
- **Estimate:** Range or proposal, assumptions, payment terms and ongoing costs.
- **Open questions:** What needs validation, who owns it and by when?

For the fictional voucher pilot, one acceptance check could be: given an agreed sample student, fee and concession, the clerk can review and print a voucher with the correct total. Test it against the school's approved sample, including an error case such as a missing fee. Agree separately how permissions and backups will be checked.

A useful pilot also needs a learning decision. For example, agree to observe two participating clerks produce a small batch of vouchers over one fee cycle, record completion and correction time, and compare it with a baseline measured beforehand. Choose a success threshold with the school before the pilot. If the workflow fails or the evidence is too thin, revise it or stop rather than calling the product validated. GOV.UK's alpha guidance similarly recommends testing the riskiest assumptions before committing to a solution. [3]

When I would not quote yet

Pause if nobody can name the first user, the decision-maker is absent, or the client wants a fixed launch date before a required integration can be checked. The same applies if the team cannot access a representative sample of the workflow, or the data and legal obligations are unclear.

The next step may be a paid discovery, a call with the actual user or an integration test. Put a limit and a deliverable on that work too. The aim is to make the next decision safer, not to turn a one-hour conversation into a promise to build everything.

*[Figure: image]***Simple wins: designing for the least technical user in the room**: People pick software they enjoy over the longest feature list. A practical guide to designing SaaS for

users who would rather not use software at all. *[Figure: image]***Code is not a business: what 20 paying customers taught me**: I built a POS SaaS, sold it to about 20 shops, then saw it solved the wrong problem. How I validate a SaaS idea now, and the checklist I use before coding.

> **Key takeaways**
>
> - Use the first call to identify one user, one problem and one complete workflow.
> - Record exclusions, manual steps and essential security alongside features.
> - Treat the estimate as conditional until the important unknowns are checked.
> - Send a one-page scope with acceptance checks and a named owner for each open question.
> - Validate with actual users before treating the outline as a final specification.

Questions readers ask

Can you scope an MVP in one call?

One call can produce a provisional scope, a list of assumptions and a plan for the next decision. Validate the risky assumptions with real users and check integrations before treating the scope or estimate as a commitment.

What should an MVP scope include?

Name the first user, their problem, one complete workflow, included and excluded work, essential security and reliability needs, acceptance criteria and a way to learn whether the product helps. Record open questions and dependencies.

Should you give a fixed price on the first call?

Give a range only when its assumptions and exclusions are clear. If the workflow or a key integration is uncertain, propose a discovery step before committing to a fixed price and deadline.

References

1. Government Digital Service. How the discovery phase works.
2. Government Digital Service. Learning about users and their needs, 2017.
3. Government Digital Service. How the alpha phase works.
4. OWASP. Application Security Verification Standard.

CITE AS

Gulzaib, M. (2026, August 20). How I scope an MVP in one call: my question list. Gul’s Notebook. https://gulzaib.vercel.app/blog/how-i-scope-an-mvp-in-one-call-my-question-list

© 2026 Muhammad Gulzaib. Quote short passages with a link and credit; please ask before republishing the full text.