Meta-Prompting for Corporate Valuation with LLMs

ApexCo workflow companion | from prompt generator to auditable DCF workflow

This handout is a companion to the **DCF Models with LLMs I** and **DCF Models with LLMs II** workflow handouts. Its purpose is not to teach another valuation method. Its purpose is to teach a **higher-order prompting method**: how to design prompts that generate better prompts for each stage of a DCF workflow. The immediate payoff is better retrieval, cleaner tables, tighter audit trails, and less generic output. The larger payoff is that students can carry one disciplined prompting system from the ApexCo mini-case into the final project without letting the model replace financial judgment.

Workflow in one sentence

Method gate first, prompt compiler second, stage execution third, audit always. A meta-prompt should first lock the valuation object and stage objective, then generate a reusable work prompt, and only then let the analyst run the task in a fresh thread with verification and revision.

1. Lock the method	2. Compile the prompt	3. Run the stage	4. Red-team the output	5. Log the work
valuation object, date, units, DCF logic, bridge policy	role, process, inputs, outputs, safeguards	architecture, facts, forecast, WACC, bridge, multiples	missing evidence, weak logic, hidden inconsistencies	Source_Log, Prompt_Log, and assumption updates

1. Why meta-prompting belongs in a valuation course

A practical definition

A **direct prompt** asks the model to do a task. A **meta-prompt** asks the model to design the best prompt for that task. In other words, the direct prompt is the immediate instruction; the meta-prompt is the prompt **compiler**. In valuation work, that extra layer matters because each stage has a different objective, a different source packet, a different output artifact, and a different failure mode.

Why valuation is a good home for meta-prompting

Corporate valuation is not one task. It is a sequence of distinct tasks: extract business architecture, build fact banks, organize history, translate evidence into assumptions, assemble the explicit forecast, estimate continuing value, estimate WACC, build the EV-to-equity bridge, cross-check with multiples, and write a memo. A single vague prompt usually fails because it blurs these stages. A meta-prompt keeps the stages separate and therefore keeps the method disciplined.

1.1 A simple ApexCo example

The weak one-shot prompt

```
Read the ApexCo packet and tell me what drives growth.
```

A stronger direct prompt

```
Using ApexCo's 2024 10-K, FY2024 earnings release, Q4/FY2024 call, and Q1 2025 10-Q,
extract the most decision-useful growth evidence for a DCF. Organize the output by
Equipment Systems, Service & Parts, and Controls Software. Return exact quotations
or hard numbers, source labels, and one possible forecast implication for each item.
Do not write a forecast.
```

The meta-prompt version

```
Design the best reusable prompt for an LLM research assistant whose job is to build
ApexCo's Growth Fact Bank. The generated prompt must: (i) define the role, (ii)
require exact quotations and source anchors, (iii) force output by segment and KPI,
(iv) separate FACT from INFERENCE, (v) forbid forecast conclusions, and (vi) add an
open-questions section if the packet is missing evidence. Return a system prompt, a
starter user message template, and a verification checklist.
```

Version	What it asks for	What it misses	Why the meta-prompt is better
Weak one-shot	A generic explanation of growth	No source boundary, no segment logic, no output schema, no audit trail	It produces prose, not an analyst-ready artifact
Stronger direct prompt	Better packet, better structure, and clear taboos	Still a one-off instruction; hard to reuse or adapt to margins, WACC, or bridge work	It improves quality, but the analyst must rewrite the task every time
Meta-prompt	A reusable prompt package for the stage	Requires one extra step up front	The extra step pays off because the same design logic can be reused across the entire valuation workflow

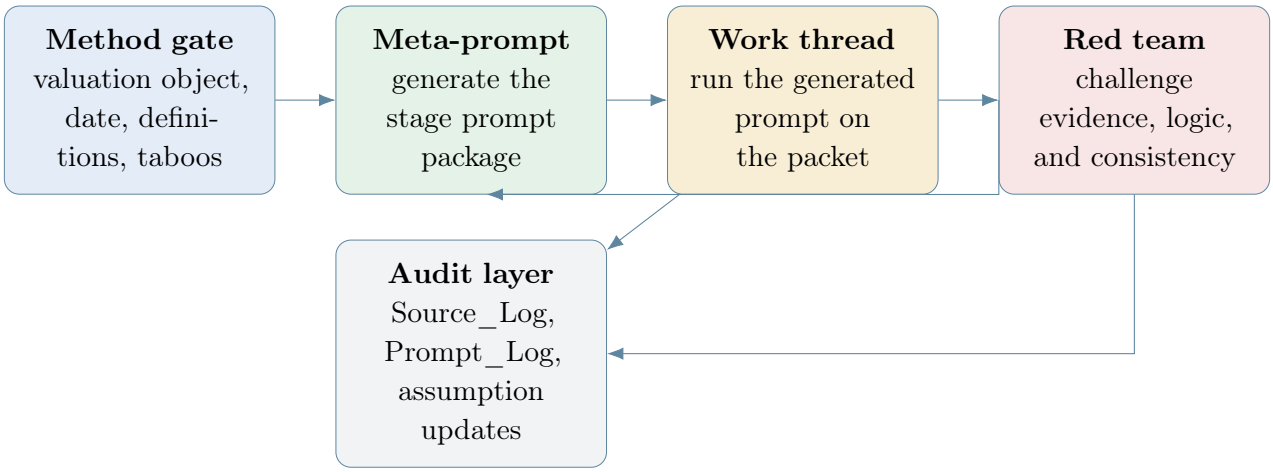
2. ApexCo refresher | the running company in the workflow

Why keep using ApexCo here?

The purpose of this handout is to teach a **transferable prompting system** without handing students a ready-made final-project solution. ApexCo therefore remains the running company. The names, engines, and KPIs below are the same as in the DCF handouts, so the meta-prompting framework connects directly to the workflow students already know.

ApexCo engine	Economic character	Key KPIs	Why the model separates it
Equipment Systems	Largest revenue engine; lower margin; more cyclical; heavier working-capital needs	Backlog, replacement demand, plant utilization, price-cost, channel inventory	It needs its own growth, margin, and working-capital logic
Service & Parts	Recurring and steadier; stronger margin quality; tied to installed base	Installed base, contract penetration, renewal rate, service revenue under contract	It stabilizes the forecast and changes the quality of FCFF
Controls Software	Smaller today but highest incremental margin; supports durability and mix uplift	Attach rate, monitoring subscriptions, retrofit modules, software mix	It explains why mature returns can remain above the cost of capital

3. Meta-prompting in one picture



What the diagram is really saying

Do not put every instruction into one giant conversation. Separate the workflow into three layers: **(i) method gate**, **(ii) prompt generation**, and **(iii) task execution**. The audit layer sits underneath the whole system. This separation is what makes LLM use feel less like improvisation and more like a professional operating process.

4. The core design framework | method gate + R-P-I-O-F

4.1 The method gate

Why the method gate comes first

Before the model writes any prompt, it should be forced to lock the valuation object and the modeling rules. Otherwise the generated prompt may quietly drift from enterprise value to equity value, from FCFF to EBITDA logic, or from DCF to generic discussion. A good method gate is short, explicit, and hard to misread.

Method gate item	What should be locked	ApexCo-style example
Valuation object	Enterprise value via FCFF or equity value via FCFE	**Use FCFF DCF and bridge from EV to common equity value**
Forecast spine	Segment logic, driver tree, and explicit period length	**Use segment engine, fact banks, Hist_Drivers, and six-year explicit forecast**
Continuing value method	Growth formula, terminal ROIC or RONIC logic, and maturity story	**Use FCFF$_{t+1}$/(WACC - g) with terminal growth tied to terminal RONIC**
Discount rate	WACC or cost of equity, and market-value weights	**Use market-value WACC for FCFF**
Bridge rule	Debt, leases, excess cash, other claims, and diluted shares	**Classify bridge items explicitly and keep lease treatment consistent**
Multiples rule	Enterprise or equity multiples and their role	**Use enterprise multiples as cross-checks; do not let multiples replace the DCF**
Audit rule	Source boundary, labels, and logging	**Use FACT / ASSUMPTION / INFERENCE labels and update Prompt_Log**

4.2 R-P-I-O-F as the design skeleton

R-P-I-O-F

A good valuation meta-prompt can almost always be decomposed into five design moves:

1. **Role**: who the model is in this stage.
2. **Process**: the ordered method it must follow.
3. **Inputs**: which packet, tables, and prior artifacts it may use.
4. **Outputs**: the exact artifact it must return.
5. **Feedback**: how the prompt should adapt after critique without restarting.

Element	What the meta-prompt should specify	ApexCo illustration
Role	Analyst assistant, research retriever, bridge auditor, memo editor, or red-team reviewer	**Growth Fact Bank builder for ApexCo** is different from **WACC evidence auditor for ApexCo**
Process	Ordered steps, method constraints, and stop rules	**Read packet -> organize by segment -> extract exact quotes -> label by KPI -> add open questions**
Inputs	Named files, source hierarchy, prior tabs, date, and known architecture	**Use 10-K, earnings release, call transcript, Q1 2025 10-Q, and the existing segment engine table**
Outputs	Tables, note format, labels, columns, and error handling	**Return one table with source, quote, classification, model implication, and confidence**
Feedback	Revision instructions and changelog logic	**If the analyst says the output is too generic, tighten to segment-specific evidence and update only affected rows**

4.3 Three additional rules that matter in valuation

Three rules students should not skip

1. **Separate FACT, ASSUMPTION, and INFERENCE.** This is the fastest way to stop quiet hallucination.
2. **Generate artifacts, not generic prose.** Tables, logs, and note-ready structures are easier to verify and easier to move into the model.
3. **Ask for one stage only.** A prompt that tries to build architecture, choose peers, forecast margins, and write a memo at once will almost always smear methods together.

5. The valuation workflow map | what the prompt compiler must produce

Stage	Valuation question	Compiled work prompt must force	Main artifact	Main risk to suppress
0. Method gate	What exactly is being valued and under what rules?	Definitions of FCFF, continuing value, WACC, bridge policy, units, and date	Method sheet or short gate note	Method drift
1. Architecture pack	What are ApexCo's real economic engines?	Segment engine table, blueprint, monetization map, KPI ledger, driver tree	Architecture pack	Generic business-summary language
2. Fact banks	What evidence matters for growth, margins, reinvestment, and continuing value?	Exact quotations, source anchors, segment or KPI tagging, and open questions	Growth, margin, reinvestment, and CV fact banks	Mixing facts with conclusions
3. Historical driver spine	What do history and segment economics say about ROIC and FCFF?	Forecast-ready tables, consistent signs, and model-ready labels	Histories and Hist_Drivers note	Pulling raw numbers with no explanation
4. Forecast assumption bridge	How should facts translate into explicit assumptions?	Range, base-case point estimate, rationale, and falsifier for each key driver	Assumption bridge table	Letting the model choose assumptions by itself
5. Explicit forecast	How do segment growth, margin, and reinvestment become FCFF?	Forecast rows, fade logic, and consistency checks across growth, investment, and returns	Forecast model build guide	Spreadsheet-only storytelling
6. Continuing value	What mature-state economics deserve to survive?	Terminal growth, mature margin, terminal RONIC or ROIC logic, and hidden-multiple check	Continuing-value note	Freezing peak conditions forever

Stage	Valuation question	Compiled work prompt must force	Main artifact	Main risk to suppress
7. WACC	What discount rate matches the valuation object?	Market-value weights, peer beta logic, cost of debt evidence, and explicit assumptions	WACC evidence pack	Mixing book and market inputs
8. EV-to-equity bridge	What sits between enterprise value and common equity value?	Debt-like claims, nonoperating assets, operating cash, dilution, and source anchors	Bridge map	Double counting or omission
9. Multiples	Do peer multiples support the DCF range?	Peer logic, enterprise multiple consistency, and conversion to equity value where needed	Multiples cross-check note	Mechanical median worship
10. Sensitivity and scenarios	Which assumptions actually move value?	Coherent scenario bundles and two-variable sensitivity design	Scenario table and sensitivity plan	Isolated one-input shocks
11. Memo and QA	Can the work survive a skeptical reader?	Thesis, exhibits, caveats, source links, and red-team issues list	Draft memo and QA sheet	Polished writing with weak evidence

6. A professional meta-prompt framework for the full workflow

6.1 The prompt package that the meta-prompt should generate

What the compiler should return

A strong valuation meta-prompt should not return only one block of instruction. It should return a **prompt package**. In practice, the package should contain four pieces:

1. a **system prompt** that defines the role and working rules,
2. a **starter user message template** that collects the stage-specific inputs,
3. an **output schema** that tells the model exactly what artifact to return,
4. a **verification checklist** that the analyst can use before moving the output into the spreadsheet or memo.

Optional add-ons are useful too: a red-team add-on, a conservative-bias add-on, and a memo-packaging add-on.

Prompt-package piece	What it does	Why it matters in valuation
System prompt	Defines role, rules, process, and taboos	Keeps the model from sliding into generic finance prose
Starter user message	Supplies packet, date, architecture, stage, and target artifact	Makes it easy to rerun the workflow on a new company or a new stage
Output schema	Names the columns, note structure, and labels required	Produces artifacts that fit directly into the workbook and report
Verification checklist	Lists the checks the analyst must apply before accepting the output	Prevents blind adoption and strengthens the audit trail

6.2 The stage brief | the variable input students should change each time

A simple working rule

The **meta-prompt should stay stable**. What changes from stage to stage is the **stage brief**. The stage brief tells the compiler what the next artifact is, which packet belongs to that artifact, what the model must not do, and what a successful output looks like.

Stage-brief field	What to write
Stage name	The exact stage, for example `Architecture Pack` or `WACC Evidence Pack`
Objective	One sentence on what the artifact must accomplish
Valuation method constraints	FCFF, continuing-value formula, WACC policy, bridge rules, or multiples rules that matter here
Authorized inputs	Source packet, prior tabs, architecture notes, market sheet, or peer list
Required output	Table names, columns, note sections, and labels
Taboos	No invented data, no conclusions beyond evidence, no peer choice without rationale, and so on
Quality checks	What the analyst will test before accepting the result

6.3 The master compiler logic

Main idea of the master meta-prompt

Instruction to the compiler: "Generate a prompt package for one and only one stage of a corporate valuation workflow. Do not do the stage yourself. Use the method gate and stage brief to produce: (i) a system prompt, (ii) a starter user message template, (iii) an output schema, and (iv) a verification checklist. Enforce source boundaries, FACT / ASSUMPTION / INFERENCE labels, open questions, and revision rules. Keep the style professional, specific, and operational."

The full master meta-prompt appears in Appendix A. It is intentionally written as a **reusable compiler**, not as a direct instruction for any one task. The student should not keep rewriting that core block.

The student should keep rewriting the **stage brief**.

7. Stage-by-stage meta-prompting for the ApexCo workflow

7.1 Stage 1 | Architecture pack

What the compiled prompt should accomplish

The architecture stage is the foundation for the rest of the workflow. The generated work prompt should force the model to identify the engines, customers, monetization logic, KPIs, and the compact driver tree that later powers the fact-bank prompts.

Stage brief ingredient	ApexCo architecture-stage version
Objective	Build the segment engine table, business blueprint, monetization map, KPI ledger, and compact driver tree
Authorized inputs	2024 10-K, FY2024 earnings release, Q4/FY2024 call transcript, Q1 2025 10-Q
Required output	Five compact tables plus an open-questions list
Main taboo	No valuation conclusions and no generic strategy language
Quality checks	Are the engines truly economically distinct? Are the KPIs specific enough to anchor later retrieval?

7.2 Stage 2 | Fact-bank retrieval

What changes here

Once architecture exists, the meta-prompt should compile work prompts that use the company's own vocabulary. The prompt package should force exact quotations or hard numbers, source anchors, topic labels, segment or KPI tags, and a short note on why the evidence matters.

Fact bank	Main retrieval handles	Required labels	Output form	Main failure mode
Growth	Backlog, replacement cycle, contract penetration, attach rate, retrofit demand	FACT / INFERENCE, segment, KPI, source	Quote bank table	Mixing temporary commentary with structural demand
Margin	Service mix, controls mix, price-cost, utilization, launch costs	FACT / INFERENCE, segment, durability	Quote bank table	Treating one-quarter noise as a permanent margin shift
Reinvestment / ROIC / FCFF	Capex cycle, inventory, receivables, automation build, D&A lag	FACT / INFERENCE, operating vs financing, timing	Quote bank table	Ignoring the link between growth and investment
Continuing value	Installed-base durability, software stickiness, mature growth, terminal return logic	FACT / ASSUMPTION boundary, terminal relevance	Short note + quote bank	Smuggling peak assumptions into terminal value

7.3 Stage 3 | Historical driver spine and driver tree

What the compiler should force in this stage

The generated prompt should ask for **forecast-ready history**, not just extracted financials. That means clean labels, clear signs, direct ties to revenue, NOPLAT, invested capital, ROIC, and FCFF, plus a note on why each line matters for the forecast.

What to lock into the stage brief	Why it matters
Explicit line list	Prevents the model from omitting a line that later breaks the ROIC or FCFF build
Model-ready naming	Makes the output easy to move into `Hist_IS`, `Hist_BS`, `Hist_CF`, and `Hist_Drivers`
Sign rule	Avoids positive / negative confusion in net investment and bridge items
Why-it-matters note	Forces the output to stay analytical instead of clerical

7.4 Stage 4 | Forecast assumption bridge and explicit forecast

The most important control point

This is the stage where students are most tempted to let the model decide the forecast. The meta-prompt should explicitly block that. The generated work prompt should ask the model to organize **ranges, logic, and falsifiers**; the analyst still chooses the final point estimate.

Compiled output block	What it should contain	Why it is useful
Range	Plausible interval for growth, margin, reinvestment, or fade speed	Prevents false precision
Base-case candidate	One defensible point estimate inside the range	Gives the analyst a clean starting point
Rationale	Why the point estimate fits the evidence	Makes the logic memo-ready
Falsifier	What evidence would make the estimate too aggressive or too conservative	Forces humility and revision discipline

7.5 Stage 5 | Continuing value

The meta-prompt should protect against terminal sloppiness

The continuing-value stage should force a mature-state story, not just a formula. The generated prompt should ask for terminal growth, mature margin or NOPLAT quality, terminal RONIC or ROIC logic, and a short explanation of why the mature economics are above, near, or below the cost of capital.

Continuing-value stage check	What the compiled prompt should ask
Growth sustainability	Is terminal growth anchored in replacement demand, installed base, or software durability rather than temporary share gains?
Return logic	Is terminal RONIC or ROIC consistent with the company's moat, mix, and capital intensity?
Reinvestment consequence	Does the output explain how much reinvestment the mature growth rate requires?
Hidden-multiple test	Does the note ask for an implied operating multiple check?

7.6 Stage 6 | WACC

Why WACC needs its own meta-prompt

WACC mixes filing-based inputs with market-data inputs, and students often confuse current leverage, target leverage, book values, and market values. A dedicated meta-prompt helps by forcing a clean evidence pack: beta logic, cost-of-debt evidence, tax-rate policy, and capital-structure weights.

WACC element	What the stage prompt should force	Typical failure if omitted
Cost of equity	Risk-free rate, beta logic, ERP, and any extra premium rule	The model smuggles in a random beta or risk premium
Beta logic	Raw beta source, peer unlevering if used, target relevering if used	Beta becomes a black box
Cost of debt	Marginal borrowing-cost evidence and tax adjustment	Students use stale coupon averages
Capital structure	Market-value weights with clear date and rationale	Students blend book and market values
Bridge consistency	Match lease treatment and debt bucket to the EV bridge	WACC and bridge tell different stories

7.7 Stage 7 | EV-to-equity bridge

The bridge stage is mostly classification

The bridge prompt should force the model to classify every valuation-relevant item into **debt-like claims**, **nonoperating assets**, **operating cash**, and **dilution or share-count items**. This is less about prose and more about a clean map with note references.

7.8 Stage 8 | Multiples cross-check

What the meta-prompt should protect here

The multiples stage is where models become mechanically overconfident. The generated work prompt should force peer rationale, comparability warnings, multiple definitions, and the conversion from implied enterprise value to implied equity value when enterprise multiples are used.

7.9 Stage 9 | Sensitivity, scenario bundles, and memo drafting

Two different roles belong here

The scenario stage and the memo stage should usually use **different compiled prompts**. One role is a scenario architect that changes connected assumptions across growth, margins, reinvestment, WACC, and continuing value. The other role is a memo writer or editor that packages the output without adding unsupported claims.

8. How to run the system in practice

8.1 A clean thread architecture

Recommended thread structure

Use different conversations or tabs for different purposes:

1. **Compiler thread**: you keep the master meta-prompt here and feed in new stage briefs.
2. **Work thread**: you paste the generated prompt package here and run the stage task.
3. **Red-team thread**: you paste the completed artifact here and ask for skeptical review.

This separation reduces instruction clutter and makes the Prompt_Log easier to maintain.

8.2 Interview mode versus fast-draft mode

Mode	Best use	Working rule
Interview mode	Early-stage work with missing context, especially architecture and bridge classification	Ask one question at a time until the packet and objective are clear
Fast-draft mode	Later-stage work when the case packet, architecture, and stage objective are already stable	Ask up to four clarifying questions at once and then draft the artifact

8.3 The revision loop

A simple revision rule

When the output is too generic, do not say only `this is generic`. Say exactly which part failed: for example, `the model mixed Service and Controls evidence`, `the bridge omitted operating cash`, or `the continuing-value note assumes peak margins`. Then ask the system to update **only the affected rows or sections**. This keeps the changelog clean.

9. Quality control, red-teaming, and logging

9.1 A compact valuation QA checklist

QA question	Why it matters
Does every important number have a source or an explicit assumption label?	Prevents quiet fabrication
Do the stage outputs match the valuation object?	Prevents DCF, multiple, and bridge logic from drifting apart
Are facts and inferences clearly separated?	Stops narrative inflation
Are growth, reinvestment, and returns connected?	Tests whether the forecast tells an economic story
Are WACC and the EV bridge consistent with one another?	Prevents double counting or omission
Do the scenarios move as bundles rather than isolated inputs?	Makes the downside and upside cases more believable

9.2 Common meta-prompting mistakes

Three common ways students misuse meta-prompts

1. **Using the compiler as the worker.** The meta-prompt should generate the task prompt, not solve the task itself.
2. **Keeping the stage brief vague.** If the stage brief does not name the artifact, the model will default to generic prose.
3. **Skipping verification because the prompt package looks professional.** A polished output is not the same thing as a verified one.

9.3 Prompt_Log fields that are especially useful

Field	Why it belongs in the log
Stage and objective	Lets another analyst reconstruct the workflow
Meta-prompt or compiled prompt used	Shows what the model was actually told
Inputs supplied	Clarifies which packet and prior artifacts were in scope
Output accepted or rejected	Makes human judgment visible
Verification action	Forces the analyst to say how the output was checked
Where it changed the model or memo	Links prompting to the final deliverable

10. Closing takeaways

The goal of meta-prompting is not to make prompting longer. The goal is to make prompting **more reusable, more auditable, and more stage-aware**. In a corporate valuation workflow, that means three practical habits:

1. lock the method before you ask for content,
2. generate prompts that are tied to one artifact at a time,
3. verify and log the output before it enters the model or memo.

Students who do this well will still use LLMs heavily, but they will use them as co-pilots rather than as ghostwriters.

11. Appendix A | Master meta-prompt for a valuation-stage prompt compiler

Reusable master meta-prompt

```
You are a Prompt Architect for Corporate Valuation workflows.  Your job is to generate
a ready-to-use PROMPT PACKAGE for one stage of a valuation process.  Do not perform the
valuation stage yourself.  Generate the prompt package only.

You will receive two inputs:  (1) METHOD GATE and (2) STAGE BRIEF.

What you must do:
1) Read the METHOD GATE and preserve its valuation definitions, object of value, units,
dates, and taboos.
2) Read the STAGE BRIEF and identify the exact artifact required.
3) Generate a PROMPT PACKAGE with four parts only:
A. SYSTEM PROMPT
B. STARTER USER MESSAGE TEMPLATE
C. OUTPUT SCHEMA
```

```
D. VERIFICATION CHECKLIST
4) Enforce these rules inside the package: no fabricated numbers; separate FACT,
ASSUMPTION, and INFERENCE; ask clarifying questions only if they are needed to proceed;
prefer structured tables over prose; include an Open Questions section when evidence is
missing; include a Revision Rule so the next draft updates only affected rows or sections.
5) Keep the style operational, precise, and professional. Do not write motivational
filler.

Requirements for Part A: SYSTEM PROMPT
- Define the role for this stage clearly.
- State the working principles and data boundary.
- State the ordered process the model must follow.
- State what the model must not do.
- Require the exact output format needed for the stage.

Requirements for Part B: STARTER USER MESSAGE TEMPLATE
- Include placeholders for company, valuation date, packet supplied, prior artifacts
available, and any stage-specific constraints.
- Include a short field for what the analyst already knows and what remains uncertain.

Requirements for Part C: OUTPUT SCHEMA
- Name every table, section, column, or label the stage should return.
- Make the output easy to move into a workbook or memo.

Requirements for Part D: VERIFICATION CHECKLIST
- Include 5 to 8 checks the analyst should run before accepting the output.
- The checks should fit the stage and the valuation method.

Output rule:
Return only the PROMPT PACKAGE. Do not do the stage work. Do not compute valuation
outputs.
```

12. Appendix B | Reusable stage-brief template

Stage brief template

```
STAGE NAME:
OBJECTIVE:
METHOD CONSTRAINTS THAT MATTER HERE:
AUTHORIZED INPUTS:
KNOWN COMPANY ARCHITECTURE OR PRIOR ARTIFACTS:
REQUIRED OUTPUT ARTIFACT:
TABOOS:
QUALITY CHECKS THE ANALYST WILL APPLY:
MODE: Interview or Fast Draft
```

13. Appendix C | A reusable red-team add-on

Red-team prompt

```
You are a skeptical valuation reviewer. Your job is not to rewrite the artifact from scratch. Your job is to challenge it. Read the artifact and identify the most important weaknesses in evidence, method consistency, assumptions, or bridge logic. Return a concise table with columns: Issue, Why it matters, What evidence would resolve it, and Priority. Do not invent new facts. Use the same valuation method and the same source boundary as the original workflow.
```

14. Appendix D | A compact Prompt_Log template

Date / stage	Prompt used	Output received	Verification action	Where it changed the model or memo
2026-xx-xx / Architecture	Meta-prompt + architecture-stage package	Segment engine table and KPI ledger	Checked against packet language and removed one generic row	Segment_Engine and Fact_Bank setup
2026-xx-xx / WACC	Meta-prompt + WACC-stage package	Beta normalization table and debt-cost evidence note	Verified peer inputs and matched debt policy to bridge	WACC tab and report exhibit