1) Executive Summary

What we're building: Earnings Call Insight (ECI) is a web app that helps users quickly understand what management cared about over time by analyzing quarterly earnings call Q&A (investor questions + management answers) and annual reports. Users choose a ticker and a time window (year + quarter); ECI retrieves the relevant files from our internal document store, routes them through multiple LLMs (e.g., GPT-5, Claude by Anthropic, Gemini by Google), and returns an insight summary: top themes, turning points, and cited evidence from the source docs.

Why now? Analysts and operators drown in text: multiple years × 4 quarters/year × long transcripts. Existing tools summarize single calls, but rarely track how focus shifts (e.g., from "growth" to "margin/FCF") with clear citations. ECI solves this by (1) structuring content around Q&A; (2) extracting themes across quarters; and (3) returning explainable, evidence-linked outputs.

What success looks like: A user can select *AMZN, 2022Q1–2023Q4* and, within seconds, see Top 5 themes with quarter-level heat, cited quotes, and detected inflection points; download/share results; and compare to peers.

2) User Research & Problem Definition

Personas (2–3)

1. **Sell-side equity analyst**
   - Needs: fast quarterly deltas; defensible theses; source citations for notes.
   - Pain: hours scanning transcripts; inconsistent tagging; misses subtle shifts.
2. **Buy-side analyst**
   - Needs: high-signal summaries before investment decisions; YoY/QoQ focus changes; risk red flags.
   - Pain: fragmented tools; manual spreadsheets and analysis; hard to compare peers.
3. **Academic researcher**
   - Needs: learn a sector's history fast; exportable graphs; glossary-level clarity.
   - Pain: steep ramp-up; "too much text, too little structure".

Current vs. Ideal Journeys

- Current way to do it: Search transcripts in database (capital IQ, company website, etc) → skim 60 - 90 min texts → CTRL+F search for keywords → paste into notes → subjective highlights → weak quarter-over-quarter continuity.
- Ideal (ECI solution): Pick ticker + time range → ECI retrieves relevant text from database → multi-LLM analysis → theme timeline + top quotes + changes over time with citations → export/share.

Problem Validation (evidence)

Validated by our team members' personal experience and lots of inconvenience we observed when reading dozens of calls per company per year. When analysts are doing equity research, most of the time is spent locating and skimming, and most tools summarize one call; It always needs trend + evidence across quarters summarized by themselves manually, because existing information platforms like Capital IQ and FactSet, don't deliver quarterly trend tracking with source-linked proof.

3) Product Vision (MVP scope & success metrics)

MVP scope (Sprint 2–3):

- Upload/store PDF/TXT earnings calls & annuals per ticker/quarter.
- Quarter-aware selection UI: `{ticker, start:{y,q}, end:{y,q}}`.
- Backend service (Django) that accepts doc path, reads text, runs multi-model LLM routing (GPT-5 / Claude / Gemini), and returns:
    - Top themes (3–5) with quarter heat;
    - Turning points analysis if any (e.g., "2023Q3: focus shifts to FCF").
    - Risks/uncertainties with evidence and citations.

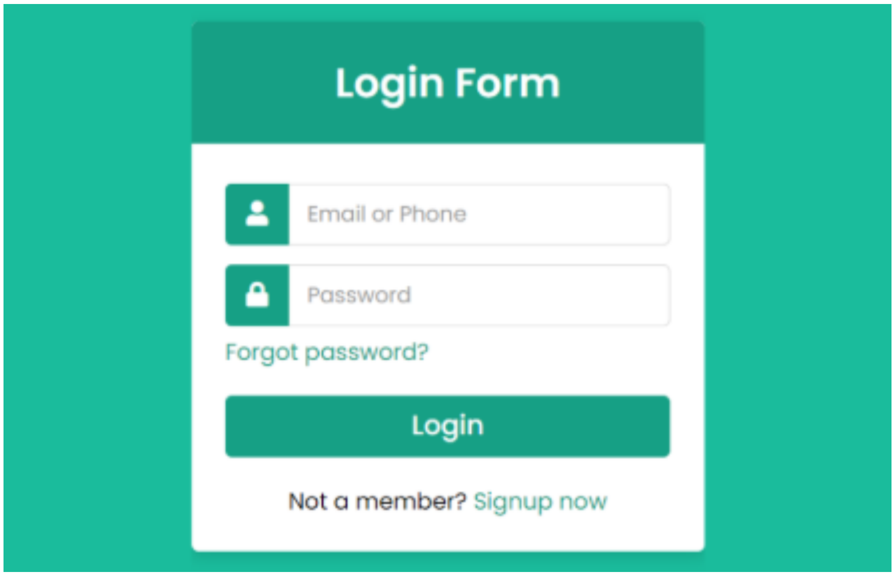
Nice-to-have (post-MVP): peer comparison; CSV/PPT export; topic alerts; quantitative KPI extraction.

Success metrics (first semester):

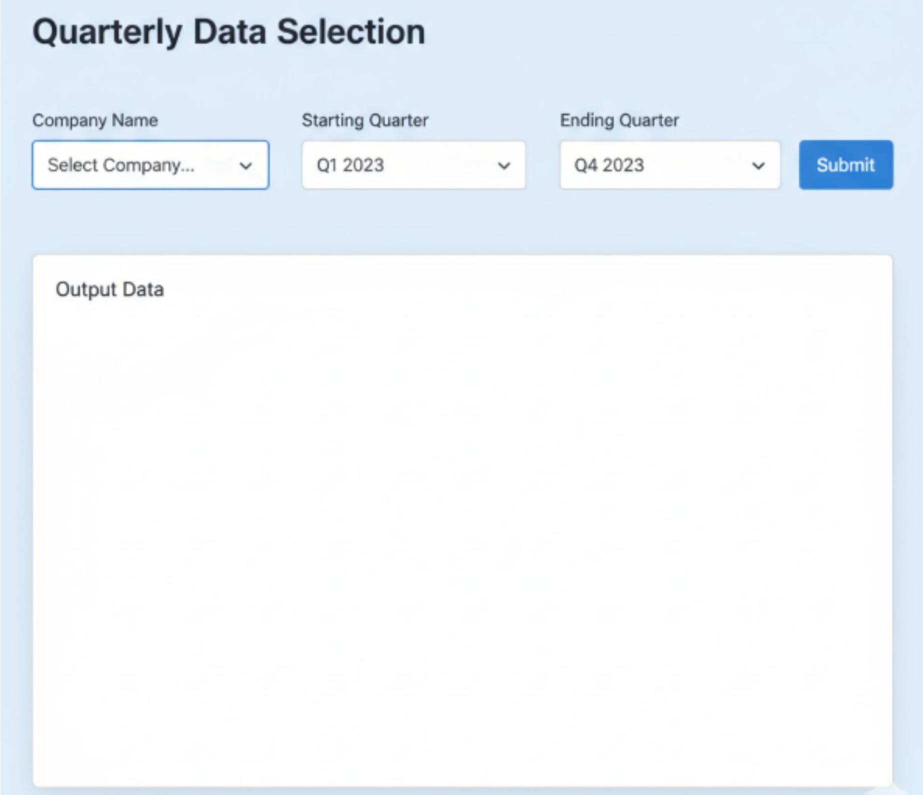
- Accuracy and user trust: users can have a smooth experience retrieving the information needed through ECI. The information can be useful and complete enough for their real-world analysis at work.
- Coverage: ≥ 2 tickers × 4 years (16 quarters) each.
- Information collection: through registration page, collect users information (name, email address) successfully
- Adoption: ≥ 10 active users (class/club partners).

4) Wireframes / Mockups

The login page will look similar to the pic below.



The page for ticker/time selection and output will look similar to the pic below.



Starting Quarter
Q1 2023

Ending Quarter
Q4 2023

5) Technical Architecture (initial decisions)

Backend:
Django (chosen for stability, batteries-included admin, easy REST endpoints).

LLM Orchestration:

LiteLLM (one API for GPT-5/4o depending on cost/time balance, Claude, Gemini).

Prompting:

Task-specific templates (Trend / Investor-Focus / Claim-Check), JSON-only outputs, evidence IDs required.

Data store:

File stored in the system folder directly given that we currently have a comparably small size of files. We can set up optional SQLite for metadata (MVP) later if we have capability.

Retrieval:

MVP concatenates targeted files; next sprint migrates to RAG (segmenting Q&A, embeddings, ANN search).

Deployment:

Dev on local/Colab; stage on Render.

6) Product Backlog Summary

- **Stories: 10**
- **Total story points:** 42 (Fibonacci)
- **Priorities:** High / Medium / Low

Examples:

Stories	Acceptance Criteria / AC	Story Points / SP	Priority / Label
As a user, I can select ticker + quarter range so that results focus on the exact period.	valid ticker; invalid quarter rejected; range inclusive; persisted in URL.	3	High
As a router service, I can pass `doc_paths[]` to the analysis API so the backend reads only those files.	rejects paths outside allowed folder; returns 400 on missing file; logs used paths.	5	High
As a user, I see Top 5 themes with quarter-by-quarter heat and one or more citations per theme.	JSON schema validated; ≥1 quote/file path per theme; handles no-evidence gracefully.	8	High

As a PM/analyst, I see turning points detected with linked quotes.		5	High
As a developer, I can switch LLM models per request.		5	Medium
As a user, I can open an evidence drawer to view quotes with file path and quarter.		5	Medium
As a system, I reject inputs without ticker/start/end/doc_paths with clear 400 errors.		2	High
As a system, I limit token usage by truncating overly long files and warn in the payload.		3	Medium
As a developer, I have a JSON-only LLM schema for themes/turns/risks.		3	High
As a user, I can export the JSON results.		3	Low
Total Story Points			

7) Sprint 2 Plans

Sprint goal: Get a working web app that can take a ticker and quarter range, run analysis, and return basic insights.

User stories selected in sprint 2: No.1- No.5 stories in Part 6.

Total story points: 26 pts

Team member assignment:
Alex Xu → front-end
Peter Yang → backend
Sarah Zhang → integration and testing
Time committed per week per member: 3-4hrs each + 3hrs together

Identified risks:
LLM API limits access; Long files may cause slow responses.

8) Team Organization

Link to team charter, roles, meeting schedule:

- Project name: Randomwalkwallstreet
- Team members: Alex Xu, Sarah Zhang, Peter Yang
- Roles: Scrum Master: rotates each sprint; Product Owner Proxy: Alex Xu; Developers: Sarah Zhang, Peter Yang.
- Communication: Email, Wechat and zoom calls; Weekly meeting

- Working Agreements: Code reviews within three days; Branch naming to be determined; Commit format short and clear
- Role Rotation: Scrum Master changes every sprint (Alex → Sarah → Peter → Alex...)
- Conflict Resolution: Talk directly and respectfully; Discuss as a team if needed; Scrum Master helps mediate.

9) Process Setup

Link to GitHub Project:

github.com/Atmoskun/randomwalkwallstreet

Link to Definition of Done:

github.com/Atmoskun/randomwalkwallstreet/blob/main/Randomwalk/docs/definition-of-done.md

Link to Definition of Ready:

github.com/Atmoskun/randomwalkwallstreet/blob/main/Randomwalk/docs/definition-of-ready.md