Research and Build

- Product Thesis
    - We are building a scalable technology platform for conducting high-stakes online exams (e.g., CAT 2025 and beyond) with a projected volume of **100k–2 million aspirants per test**.
    - Each test will:
        - Consist of **50-300 MCQs**
        - Have each question tagged to **one of 19 possible subjects**
        - Be time-bound: **3-hour duration**
        - Have a taker volume ranging from **10,000 to 1,000,000+ aspirants per test**
        - Be used across **100K–1M such test instances annually**
    - Your Objective
        - You are expected to design and prototype the **backend foundation** for this platform with a focus on **performance, accuracy, and scalability**. This involves tackling three core problems.
    - Problem 1: Efficiently Storing User Submissions
        - Design a schema and submission workflow that:
            - Stores answers submitted by each user for each question
            - Tracks the correctness of the answer (correct: +5, wrong: -1)
            - Supports **subject-wise scoring** for each test taker
            - Can scale to tens of millions of records without bottlenecks
        - *Key Expectations:*
            - Document schema design (MongoDB collections)
            - FastAPI routes like `POST /submit-answer`
            - Indexing strategies for performance
            - Efficient querying for result generation
    - Problem 2: Publishing Final Test Results (Ranks & Percentiles)
        - Once the exam concludes, the system must:
            - Calculate **overall score** and **subject-wise scores** for all users
            - Compute:
                - **Ranks**
                - **Percentiles**
                - **Subject-wise percentiles**
            - Efficiently publish the final leaderboard for 1M+ users
        - *Key Expectations:*

- Algorithm/logic to compute percentiles & ranks at scale
- Trade-offs between pre-computation vs on-demand generation
- Thought process around caching, async jobs, data structure choices

- Problem 3: Predictive Results for Future Mock Test Takers
  - After the real exam ends and the result is published:
    - Future aspirants (e.g. CAT 2025 aspirants) will take **mock tests** modeled after CAT 2024
    - For each mock test user, the system must provide:
      - **Predicted Rank**
      - **Predicted Percentile**
      - **Predicted Subject-wise Percentile**
  - 💭 *What we're looking for:*
    - Approach to compare mock user's score with real test result distribution
    - Insight into how to model predictions using existing data
    - Thoughts on real-time prediction performance vs accuracy tradeoffs
- Development Expectations
  - You can use **FastAPI** (or any equivalent Python framework) to:
    - Expose REST APIs
    - Use an MongoDB DataBase
    - Show modular, readable, and scalable code structure
- Bonus (Not Mandatory but Appreciated)
  - Rate limiting or anti-cheat mechanisms
  - Basic auth or user ID simulation