# **Quick Reference Guide - Scoring Refactor Assignment**

## **🎯 Mission in One Sentence**

Extract all scoring logic from the 400+ line processSurveyResponse method into two clean services WITHOUT changing any algorithms.

## **📁 What Goes Where**

### **Into CentralScoringEngine:**

* getFixedMcqQuestionScore() - for "Never/Always" questions
* getVariableMcqQuestionScore() - for company-specific answers
* isTypeOneQuestion() - helper method
* Gemini AI scoring logic (the text answer scoring)
* Any logic that calculates a single question's score

### **Into ScoreCompositionService:**

* Standard pillars aggregation (values, culture, mindset, etc.)
* Dynamic pillars aggregation (custom company pillars)
* Z-index calculation for mindset
* calculateMatchScore() method
* Final application score calculation (survey + CV score)
* All the answer array merging logic

### **Stays in SurveyResponseService:**

* Application/Candidate linking
* Database saves
* External service calls (Gemini feedback, ATS push)
* Main orchestration flow

## **🤖 AI Prompting Strategy**

### **Phase 1: Analysis**

"Analyze this SurveyResponseService class. Create a detailed map showing:

1. All methods/logic related to scoring individual questions

2. All methods/logic related to aggregating scores into pillars

3. The main orchestration flow

Group related code sections and note dependencies."

### **Phase 2: Extraction**

"Help me extract all individual question scoring logic into a new

CentralScoringEngine service. The service should:

- Have a clean interface with a calculateQuestionScore method

- Include all the logic from [list methods]

- Maintain exact same behavior

- Use proper Spring service annotations

- Include dependency injection for GeminiService"

### **Phase 3: Testing**

"Generate comprehensive unit tests for this CentralScoringEngine method

using JUnit 5 and Mockito. Include:

- Tests for fixed MCQ scoring (Never/Always pattern)

- Tests for variable MCQ scoring (company values matching)

- Tests for text-based Gemini scoring

- Edge cases: null inputs, empty arrays, missing correct answers

- Proper mocking of GeminiService"

## **✅ Deliverables Checklist**

### **Code Files**

* [ ] CentralScoringEngine.java (interface)
* [ ] CentralScoringEngineImpl.java
* [ ] ScoreCompositionService.java (interface)
* [ ] ScoreCompositionServiceImpl.java
* [ ] SurveyResponseService.java (refactored)
* [ ] CentralScoringEngineTest.java
* [ ] ScoreCompositionServiceTest.java

### **Documentation**

* [ ] AI\_WORKFLOW.md with:
  + [ ] Tools used
  + [ ] Process breakdown
  + [ ] THE killer prompt (with explanation)
  + [ ] Challenges faced



## **📊 What Gets You Points**

### **High Value Actions:**

* **Document a genuinely useful "killer prompt"** (15 points!)
* **Show iterative AI usage** (not just one-shot)
* **Clean separation of concerns** (major design points)
* **Working tests with good coverage** (20 points total)

## **🚀 Final Tips**

1. **Read first, code second** - Understand the flow before extracting
2. **Let AI do the heavy lifting** - Use it for analysis, boilerplate, tests
3. **Document as you go** - Keep notes for AI\_WORKFLOW.md
4. **Focus on the refactoring** - Don't get distracted by other improvements
5. **Test your extraction** - Make sure it still works!

Good luck! 🎯