PRD: Coursera “Adaptive Assignment Coach” (v1.0)

Owner: Aman Dixit

Development POC: Aman Dixit

Design POC: Aman Dixit

Marketing POC: Aman Dixit

Last Version Edited: March 15

1. Executive Summary

The **Adaptive Assignment Coach** is an in-context AI tutoring interface designed to reduce learner attrition and academic dishonesty. Unlike generic LLM wrappers, this coach uses a **Socratic pedagogical model** to provide "just-in-time" scaffolding. Analyzing a learner's partial attempt against a canonical solution graph provides "nudges" rather than answers. The goal is to transform the "I'm stuck" moment from a point of abandonment into a deeper learning opportunity, directly supporting Coursera’s mission of providing universal access to world-class learning.

2. Problem Statement & User Needs

2.1 The P0 Problem

Learners in technical courses reach a "Cognitive Wall" during graded assignments. Current support (forums/generic hints) is asynchronous and non-personalized. This leads to two negative outcomes: **Churn** (dropping the course) or **Integrity Violations** (copying solutions from external sites like Chegg/Stack Overflow).

2.2 User Personas

Persona	Needs	Pain Point
Priya (Career Switcher)	Scaffolding and confidence building.	"I understand the video, but the blank code editor is terrifying."

Mark (Upskiller)	Efficient debugging and syntax sanity checks.	"I have the logic right, but I've been stuck on a 'NoneType' error for two hours."
Course Author	Scalable student support without compromising rigor.	"I can't manually hint 10,000 students, but I don't want AI to give them the answers."

---

3. Market & Competitive Analysis

- **Target Market:** STEM and Professional Data/Dev Specializations (highest drop-off rates).
- **Competitors:** * *Codecademy:* Offers "Get Unstuck" but often reveals the full solution diff, which reduces cognitive load too much.
    - *Khan Academy (Khanmigo):* Strong Socratic approach but lacks the deep integration into Coursera's specific "Canonical Solution Graphs."
- **Differentiation:** Our coach doesn't just "guess" based on the prompt; it compares the learner's AST (Abstract Syntax Tree) or logic to a pre-defined solution path provided by the instructor.

---

4. Product Goals & Success Metrics

4.1 Primary Objectives

1. **Retention:** Increase assignment completion rates by 15% in pilot courses.
2. **Integrity:** Reduce the incidence of "perfect first-time submissions" after long periods of inactivity (proxy for external copying).
3. **Efficiency:** Reduce the average time from "First Attempt" to "Passing Grade" without increasing "Hint Abuse."

4.2 Key Performance Indicators (KPIs)

- **Stuck-to-Submit Rate:** % of users who submit an assignment within 24 hours of clicking "I'm Stuck."
- **Hint Efficacy:** % of users who pass an assignment after < 3 AI interactions.
- **Practice Set Conversion:** % of users who engage with the "Generate 3 Similar Problems" feature.

---

5. User Stories & Requirements

5.1 User Stories

- **US.1:** As a learner, I want the coach to rephrase the prompt so I can ensure I haven't misinterpreted the requirements.
- **US.2:** As a learner, I want the coach to tell me *where* my logic failed (e.g., "Line 14 handles the loop incorrectly") without writing the code for me.
- **US.3:** As a learner, I want to see a similar example so I can map the concept to a new context.

5.2 Functional Requirements

- **FR.1 (The Guardrail):** The system must strip any "Full Solution" strings from its output.
- **FR.2 (The Logic Check):** The system must identify specific divergence points between the user's code and the instructor's solution graph.
- **FR.3 (Practice Mode):** The system must generate ungraded, auto-grading variants of the current problem using distinct variables/scenarios.

---

6. User Experience & Design

- **Entry Point:** A "Need help?" floating action button (FAB) or sidebar toggle located near the "Submit" button.
- **The Side Panel:** A non-intrusive drawer that maintains the context of the code editor.
- **Feedback Loops:** Visual indicators (e.g., highlighting a specific line of the learner's code in the side panel) to provide spatial context.

---

7. Technical Considerations

7.1 Architecture & Models

- **Model:** Tutor-specific LLM (e.g., Gemini 1.5 Flash for speed or GPT-4o for complex reasoning) with a heavy **System Prompt** centered on Socratic methods.
- **Canonical Solution Mapping:** The system ingests the instructor's solution and converts it into a "logic flow."
- **State Management:** The Coach must maintain the history of the current assignment session to avoid repetitive hints.

7.2 Security & Compliance

- **Data Privacy:** User-pasted code must be anonymized before being sent to the LLM API.
- **Abuse Prevention:** Rate-limiting hints to 5 per assignment per hour to prevent "brute-forcing" the AI for answers.

---

8. Rollout Strategy

- **Phase 1 (Alpha):** Internal testing with 50 Coursera employees on "Python for Everybody."
- **Phase 2 (Beta):** A/B test on 10% of learners in 3 high-traffic technical Specializations.
- **Phase 3 (GA):** Full rollout across all auto-graded coding and math assignments.

---

9. Risks & Mitigations

Risk	Impact	Mitigation Strategy
Hallucination	High	Ground the AI in the "Canonical Solution" only; restrict its "knowledge" to the specific assignment parameters.
Hint Abuse	Medium	Implement a "Hint Credit" system or cooldown timer.
Model Latency	Low	Use a smaller, faster "Flash" model for simple task clarification; reserve larger models for code analysis.

---

10. Timeline & Milestones

- **Week 1-4:** Prompt engineering & "Canonical Solution" ingestion engine development.
- **Week 5-8:** UI/UX Side-panel development and integration with Coursera's Judge0 (or equivalent) grader.
- **Week 9:** Beta Launch & Data Collection.
- **Week 12:** V1.0 General Availability.