Hack-A-Thon - Neural Pioneers - Initial Plan

NOTE: This plan might change from our future conversations or during implementation, this is just to give an idea about what we will be doing in this project.

Project Summary

- Project Name: **NeuroLearn**
- Planned technologies to use: LangChain (for basic tools), LangGraph (for orchestration), OpenAI API (paid, I've $3.45 right now, and we'll use cheap and smaller LLMs), StreamLit (for creating a website in like 1 hour with just pure python, no HTML, CSS, JS needed)
- Core Idea: To be different from everyone else's thinking, we will be building an Agentic RAG system.

---

1. Agentic Tutor + Diagnostic Explainer

- Since we only have to create a Prototype, we'll just download a NCERT book for possibly 8th class or anything for our demo dataset. We can build a data scraping pipeline for NCERT but it is not relevant and can take so much time. Following is the main pipeline:
- Instead of just showing *"You're weak in retention"*, create an **interactive agent** that *explains to the student why* they struggled.
- Example:
    - Sanga keeps failing application-based problems, the agent shows *"You're great at recalling formulas but struggle when they appear in word problems. Let's practice with step-by-step real-world examples."*
- Use **LangGraph** to orchestrate:
    - **AssessmentAgent**: runs adaptive tests
    - **DiagnosticAgent**: analyzes responses to map weaknesses to fundamentals
    - **TutorAgent**: generates practice tailored to that weak spot
- Don't just feed random harder/easier questions.
- Use **OpenAI API + LangChain** to **generate custom practice problems** tied to the detected weakness.

- Example:
    - If a student struggles with application, generate real-world context problems (e.g., “A train leaves Jaipur at 7 AM…”).
    - If they struggle with listening/grasping → generate **hinted or scaffolded problems** (step-by-step cues).

This shows creativity + real AI integration, not just static questions.

2. Categories of possible student difficulties

- **Listening**: errors in easy direct-recall Qs given right after explanation
- **Grasping**: errors in basic concept questions
- **Retention**: errors in repeated Qs after delay
- **Application**: errors in mixed/word-problem Qs

3. Gamified Learning Journey

Just simple prompt based gamification of learning, probably giving students badges like “Master Navigator”, “Explorer”, “Apprentice”, based on their learning journey to engage students into learning as a game. And also increase/decrease the level of students' understanding based on the answers.

4. Focus on class of student

In the problem statement, it says about students of class 8. So we should also take input about the student's class. And tailor the responses based on that specific class and age group. Doesn’t matter that much right now, since the data is static.

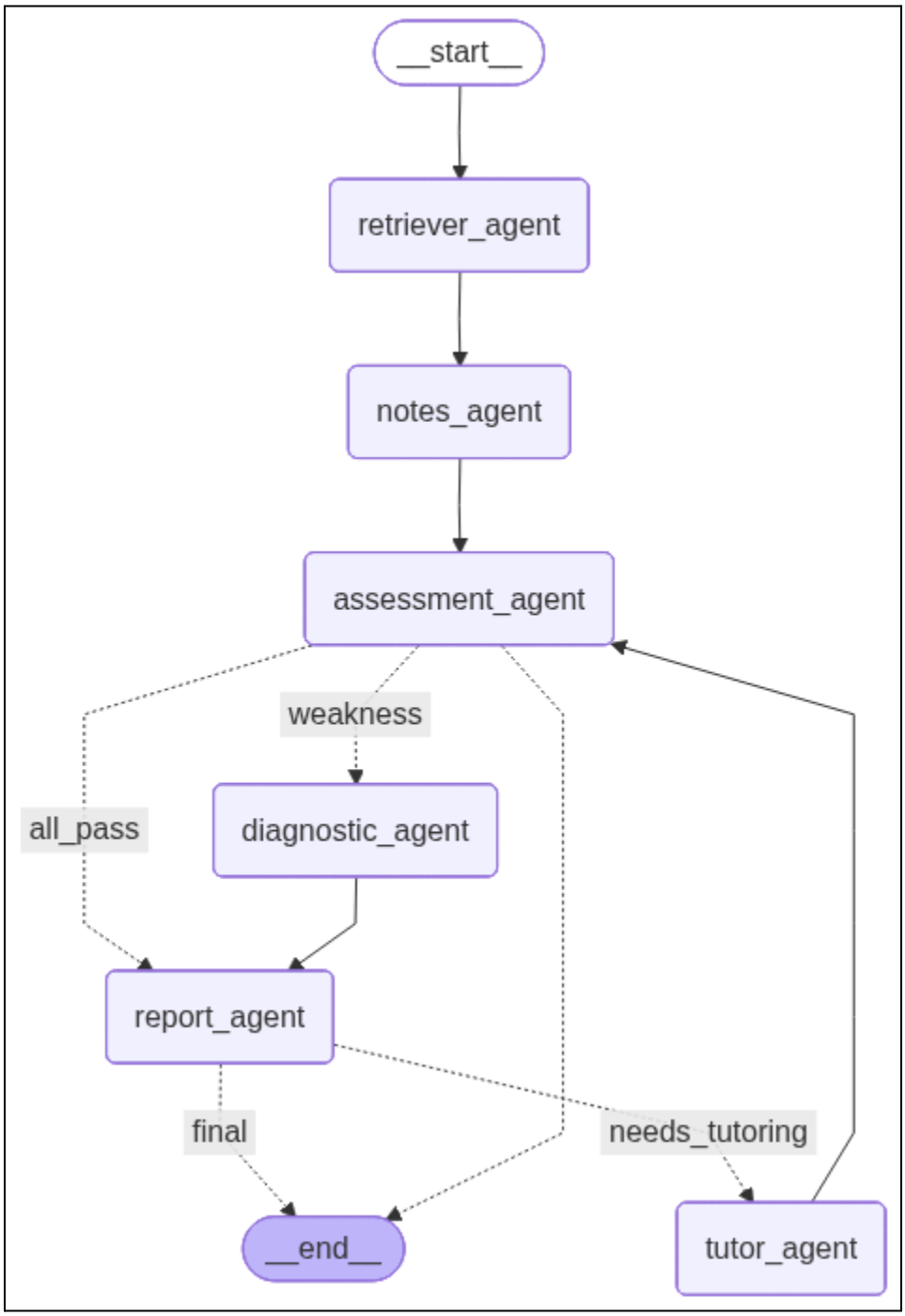
5. Parent & Teacher Friendly Dashboards

Most teams will build a student-facing prototype.
Stand out by also including **separate views**:

- **Student View**: personalized practice, gamification (our main pipeline)
- **Teacher View**: diagnostic summary per student. (a single agent (ReportAgent) will handle this by taking some kind of input and then generate a teacher friendly report).
- **Parent View**: simplified insights (not technical, but actionable, e.g. “Encourage Sanga to explain problems aloud; it builds retention”, also generated by ReportAgent in a single LLM call)

In the above points, 3, 4, 5 are really easy and are just some minor improvements, finishing touches and style changes so that we could stand out.

Initial Architecture



Workflow

1. Retriever agent will retrieve relevant data from NCERT API. But in our case we will build a demo data with only 2 math topics and 2 science topics. So no actual data retrieval. We can use 1-2 topics for training and others for testing.
2. Then the notes agent will create notes for the student to study. Students can spend any amount of time studying them, AI will only proceed when the student presses the next button in the UI.
3. The assessment agent will initially create 20 questions based on 4 categories: Listening, Grasping, Retention, and Application. Students can take any amount of time completing the quiz (or we can add a timer) the AI will only proceed when the quiz is submitted.
4. Then the assessment agent will check the answers. If there is any kind of weakness in any of the 4 categories, then it will go to the diagnostic agent.
5. The diagnostic agent will diagnose what is the exact problem with the student.
6. Then it will go to the report agent to create a report for the student's status.
7. Then the report agent will stir towards the tutor agent, and the tutor agent will resolve what the student has done wrong, why it was wrong, explain how to correctly solve it, give some tips, etc. Students can spend any amount of time on the tutor notes.
8. After this the assessment agent will again create 5-10 questions based on the diagnosis and tutor agent’s notes.
9. This cycle will repeat itself until the assessment agent says that the student has finally learnt the topic for all the 4 categories.
10. The report agent will generate some final reports and the flow will end.