Local LLM - Student Facing Rubrics

How Student will be Graded

Your grade will be based on both your code implementation and your reflection/reporting.
Some parts will be automatically graded (your code must run correctly), while others will be manually graded (clarity of your report, depth of reflection).

Section	What's Expected	Weight
Environment Setup & Dependencies	Set up Python environment correctly, install all required libraries, place models properly, and ensure the model loads.	15%
Core Functions Implementation	Implement all required functions (chunk_text, embeddings, FAISS, search, run_llm`). Functions should return correct results and demonstrate your understanding.	30%
PDF Context Integration	Successfully extract text from the PDF, generate embeddings, and use them to improve query responses. Show differences between queries with and without PDF context.	20%
Reflection & Reporting	Submit a short (1–2 pages) report that: 1. Summarizes your setup steps 2.Describes errors and how you solved them 3.Compares results with and without context 4.Reflects on how LLMs can be applied in healthcare.	20%
Submission Quality	Your Jupyter Notebook runs end-to-end without errors, includes clear outputs, and your report/screenshots are well-organized and concise	15%

Notes for Students

Autograding: Many functions will be tested automatically (e.g., chunking, embeddings, FAISS index). If your notebook doesn't run, you'll lose those points.

Manual review: Reports, reflections, and the clarity of your responses will be graded by the professor/TA.

Report requirement: 1–2 pages, concise and structured. Include examples of queries (with and without context).

Evidence: Include screenshots or copied outputs from your notebook showing that your code works.

Pro tip: Make sure your functions run as expected with simple test cases before moving to the full PDF workflow.

Local LLM - Professor Facing Rubrics

How to Grade

Grade only what's demonstrably shown: code that runs end-to-end, clear outputs/screenshots, and a concise report. Combine autograder results (does it execute and produce required artifacts?) with manual judgment (clarity, correctness, depth of analysis). Apply the four levels consistently—Exceeds / Meets / Approaches / Missing mapped to 100 / 75 / 50 / 25% of each criterion's weight—and favor reproducibility: no missing files, brittle paths, or hidden dependencies. If evidence is missing or unverifiable, assign the lower level.

Section	Criterion	Weight	Excellent (100%) – Exceeds Expectations	Proficient (75%) – Meets Expectations	Basic (50%) – Partially Meets Expectations	Inadequate (25%) – Does Not Meet	Autograde Possible	NBGrader autograde reasoning	Comments	Shreya's execution
Environment Setup & Dependencies (15%)	Python env & libraries	5 pts	All dependencies installed; imports run without error.	All dependencies installed; imports run without error.	Minor missing library/import, easily fixed.	Environment fails to run.	Yes	True, assert statement in a try/except block	Check for import errors/ No errors	No errors(100)
	Model download & placement	5 pts	Both models correctly placed and accessible in /labs/local_llm.	Models downloaded but misplaced/misnamed.	Only one or incomplete model.	No usable models found.	Yes	False, assert on filepath but system dependant	Submission is a notebook, it is not possible to check for the correct folder of download in general but the next step implies this step	Couldn't find the path of model(it was "./" i.e, current folder) (100)
	Model loading	5 pts	Model loads without error; run_llm("Hello") produces valid coherent response.	Loads with minor warnings but produces output.	Loads partially with errors in generation.	Fails to load or crashes.	Yes	True, implies above two rubrics	If this is possible then all above steps were executed correctly	Proof of model execution (100)
Core Functions Implementation (30%)	chunk_text()	5 pts	Correct chunking & overlap; outputs usable segments.	Minor overlap/size errors but mostly works.	Inconsistent splitting or incorrect overlap.	Missing or non-functional.	Yes	True, assert on function(unit testing style)	Need clear definition of overlap and size values, or the expected number of chunks	Implementation and usage of the function verified (100)
	llama_embed_text()	5 pts	Returns correct embeddings as np.ndarray with proper shape.	Returns embeddings but with inconsistent shape/type.	Embeddings returned but with evident errors.	Missing or fails completely.	Yes	True, assert on function(unit testing style)	Quality is ambiguous, need clear definition of "proper" shape and "consistent" shape/type	Implementation and usage of the function verified (100)
	Embedding loop / FAISS index	5 pts	All chunks embedded; FAISS index created and populated.	Most chunks embedded; index partially populated	Few chunks embedded or index corrupted	No embeddings or FAISS integration.	Yes	True, assert on function(unit testing style)	This can be part of previous rubric testing	Implementation and usage of the function verified (100)
	search_similar_chunks()	5 pts	Retrieves top-k relevant chunks with clear similarity logic.	Retrieves chunks but sometimes irrelevant.	Results mostly irrelevant to query.	Function missing or fails.	Yes	Partially true, assert on function possible. relevance not	Relevance needs to be defined but still hard to detect with AI(Maybe vector distance?)	Implementation and usage of the function verified (100)
	run_llm()	5 pts	Generates coherent, relevant responses to test prompts.	Responses mostly coherent; minor weaknesses.	Incomplete or inconsistent responses.	No meaningful response.	Manual	Partially True, assert on function possible, meaning is not	Meaning and consistency need to be defined based on the query in general or a predefined test query needs to be provided	Implementation and usage of the function verified (100)
	run_llm_with_pdf_knowledge()	5 pts	Effectively integrates retrieved context in responses.	Context partially integrated; some relevance.	Weak integration; context barely affects answer.	No integration attempted.	Manual	Partially True, assert on function possible, meaning is not	This is dependant on previous function, need manual + tool grading	Implementation and usage of the function verified (100)
PDF Context Integration (20%)	PDF extraction	5 pts	>500 chars extracted accurately from PDF.	Text extracted but slightly incomplete.	Only partial/limited extraction.	Extraction fails or missing.	Yes	Partially True, assert on function possible, completeness is not	If only number of words, then it is possible with a tool, if not, completeness should be defined, Can be covered in function test of run_test_with_pdf()	Implementation verified (100)
	PDF embeddings	5 pts	Embeddings generated for all text chunks.	Most embeddings generated successfully.	Partial embeddings only.	No embeddings created.	Yes	True, assert on return value	Can be covered in function test of run_test_with_pdf(), hard to validate manually	Verified based on final output (100)
	Query without context	5 pts	Shows LLM's baseline knowledge limits clearly.	Some limits visible but inconsistent.	Results unclear about LLM's limitations.	Misleadingly confident or missing comparison.	Manual	False, can write assert statements based on PDF knowledge	Comparison with next rubric	Verified the difference (100)
	Query with context	5 pts	Responses significantly improved with PDF context.	Partial improvement visible.	Limited or unclear improvement.	No observable improvement.	Manual	False, can write assert statements based on PDF knowledge	Comparison with prev rubric	Verified the difference (100)
Reflection & Reporting (20%)	Setup summary	5 pts	Clear, complete, step-by-step documentation.	Mostly clear; minor gaps.	Limited detail/clarity.	Missing or very unclear.	Manual	False, an LLM can give a detailed report based on clearly defined rubrics	Prompt engineering	Clear and concise report (100)
	Difficulties/errors	5 pts	Thorough documentation of errors & solutions.	Documents errors but not all solutions.	Mentions errors with no resolution.	No discussion of errors.	Manual	False, an LLM can give a detailed report based on clearly defined rubrics	Prompt engineering	Clear and concise report (100)
	Query comparison	5 pts	Strong analysis of context vs. no-context differences.	Adequate comparison; some insights.	Weak or superficial comparison.	No comparison provided.	Manual	False, an LLM can give a detailed report based on clearly defined rubrics	Prompt engineering	Clear and concise report (100)
	Insights & applications	5 pts	Deep reflection on LLM limitations & healthcare use cases.	Some relevant insights.	Surface-level or generic insights.	No meaningful reflection.	Manual	False, an LLM can give a detailed report based on clearly defined rubrics	Prompt engineering	Clear and concise report (100)
Submission Quality (15%)	Notebook	5 pts	Runs end-to-end with clean outputs; reproducible.	Minor execution/formatting issues.	Multiple errors, requires fixing to run.	Notebook incomplete/unusable.	Yes	Partially True, execution order not possible to verify	Execution order should be verified manually	Verified execution order (100)
	Screenshots/outputs	5 pts	Clear, labeled screenshots/outputs included.	Some outputs missing/unclear.	Outputs incomplete or poorly presented.	No outputs/screenshots.	Manual	Partially true, LLM can verify output, completeness needs manual intervention	No Screenshot is a choice since this assignment requires a fully executed notebook	Clearly defined (100)
	Report formatting	5 pts	Well-structured, concise, within 1–2 pages.	Clear but slightly long/short.	Formatting issues or unclear writing.	Report missing.	Manual	Partially true, LLM can verify contents	Formatting needs manual intervention	Clear format (100)

Performance levels map to:

100% = full 5 pts
75% = 3.75 pts
50% = 2.5 pts
25% = 1.25 pts