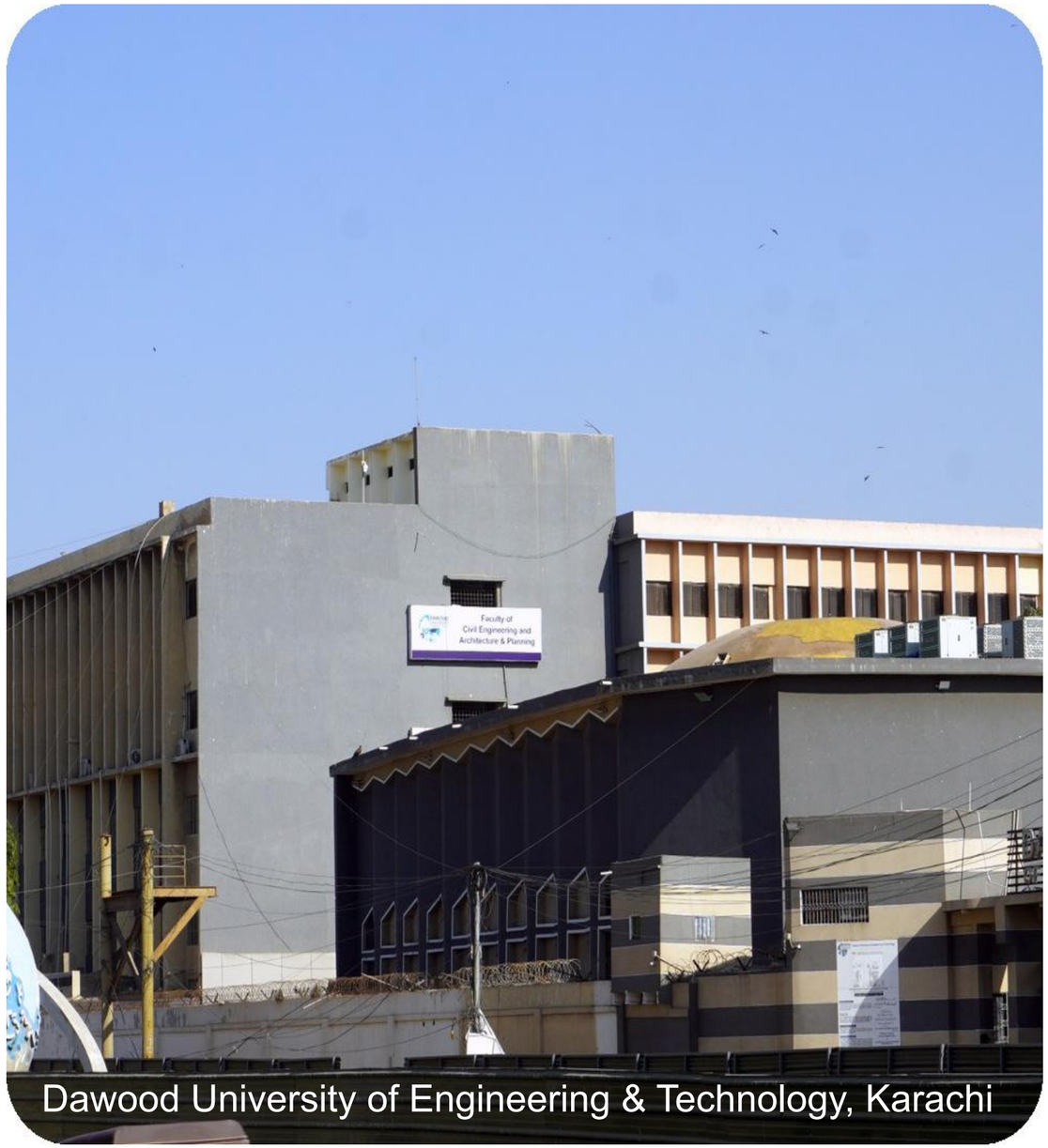
Generative AI, LLMs & Agentic AI

Your Career Roadmap

M. Huzaifa Shahbaz

Senior AI Engineer · 23 November 2026

mhuzaifa.com/duet-genai



Dawood University of Engineering & Technology, Karachi

ABOUT ME

M. Huzaifa Shahbaz

Senior AI Software Engineer

Digital Dividend Global · Karachi, Pakistan



Nearly 7 years working with AI

More than 4 years professionally

I build AI agents, core AI models, search tools and the backends that run them.

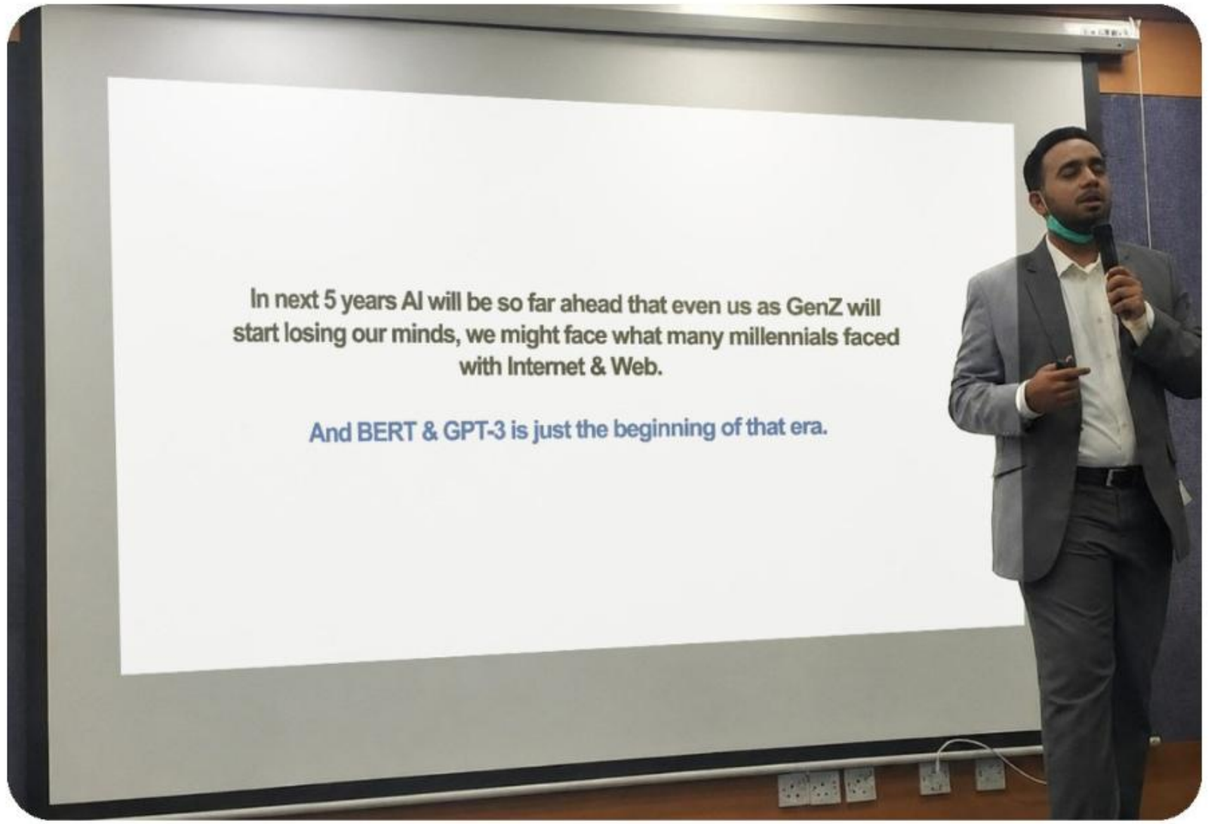


Speaking at DHA Suffa University

@mhuzaifadev · mhuzaifa.com

THE AGE OF AI / 2021

Five years ago, I said this.



Bashir A Malik Auditorium, SSUET · 2021

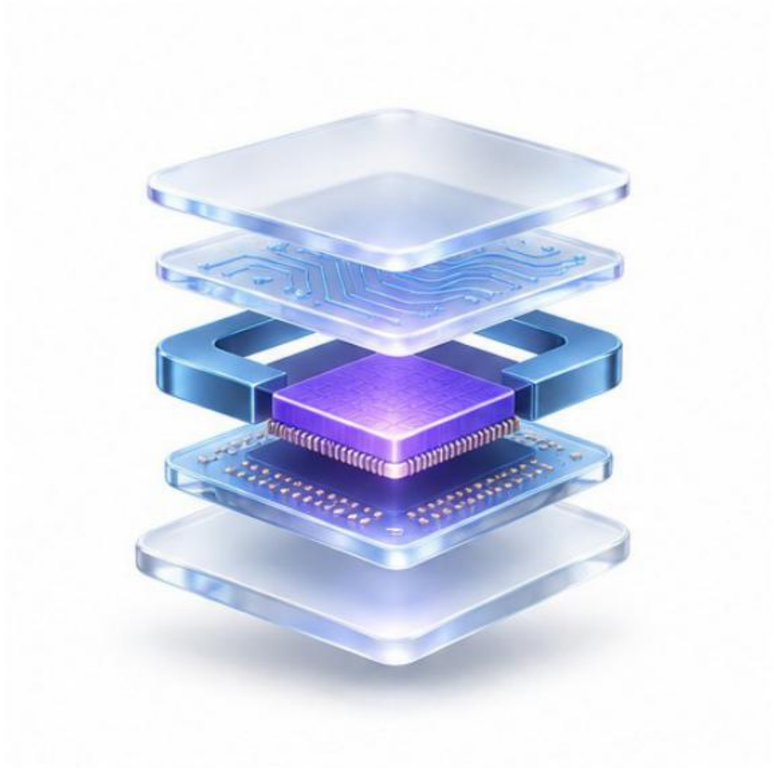
FROM MY ORIGINAL SLIDE

“And BERT & GPT-3 is just the beginning of that era.”

I thought AI might catch Gen Z by surprise the way the web caught many millennials.

Five years on, that feels close.

Where does the model run?



In the cloud

A hosted API lets you pay for the tokens your app uses.

In less memory

Quantization stores weights with fewer bits. Test the quality cost.

On the device

Chips such as Apple M4 include a Neural Engine for AI workloads.

Which have you used? Which have you built with?

What if you could buy the agent once?

~$200

One-time license

My bet for the next 18 months: some specialized AI tools will sell a version you can run on your own computer.



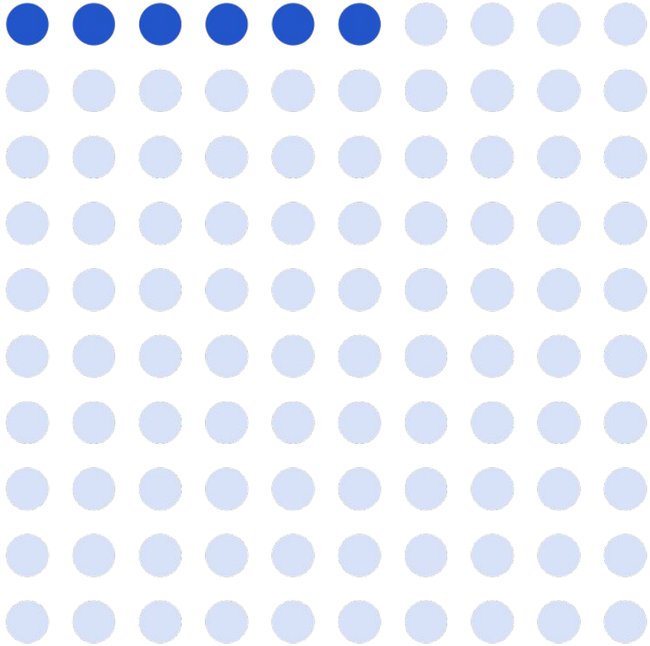
Illustrative price in USD. Hardware, support and upgrades are separate.

Myth: AI will replace developers

6%

Entry-level roles

in a sample of 3,449 AI engineer postings.



Myth: agents will run everything

>40%

of agentic AI projects will be cancelled by the end of 2027, Gartner predicts.

- High cost
- Unclear value
- Weak risk controls

Gartner forecast · June 2025

What survives the next model release?

Give two models the same 50 campus questions before you switch.

Correct answers

Does it cite the right page?
Does it flag a missing source?

Response time

Measure time to first token
and the slowest 5% of replies.

Cost per task

Count retries and tool calls.
Compare cost per success.

Release decision Ship the change only if it clears the agreed thresholds.

A raw LLM is a brilliant intern with no memory and no hands

“What’s the fee for Batch 25 and is the AI elective still open?”



Layer 1 → the LLM guesses

Campus Assistant

“The fee is PKR 80,000.”

Wrong fee. No university source.

How an LLM predicts the next token

Follow the prompt “The fee is …” through a decoder-only transformer.

Tokenize

Text becomes token IDs.
“The” | “ fee” | “ is”

Embed

Turn IDs into vectors.
Encode token positions.

Mix context

Causal attention reads earlier tokens. An MLP transforms each vector.

Decode

Logits score each token.
Softmax gives probabilities.
Pick one and repeat.

TRAINING

Adjust weights to predict the next token.

INFERENCE

Reuse the weights to generate a continuation.

Layer 2 → context engineering decides what it sees

CONTEXT WINDOW

- Instructions
- Examples
- Retrieved data
- Memory

Give it the right three pages.

Layer 3 → RAG turns it into an open-book exam



PKR 65,000

Sample prospectus · Fee schedule, page 4

Illustrative campus data

Campus Assistant

How RAG finds the right fee

The index is built when documents change. The answer is built when a student asks.

INDEX THE DOCUMENTS

Parse

Keep headings, tables and page numbers.

→

Chunk + embed

Split by section. Create a vector per passage.

→

Store

Index text, vectors, access and version.

ANSWER A QUESTION

Retrieve

Match keywords + vectors. Filter by permissions.

→

Rerank

Pick passages that address this batch and fee type.

→

Generate + check

Answer from the passages. Link page 4 or abstain.

Illustrative campus data · Test retrieval with page hit@k and test generation with answer and citation checks.

Layer 4 → tools and MCP give it hands



Model Context Protocol

A common connector for tools.

Check live elective seats

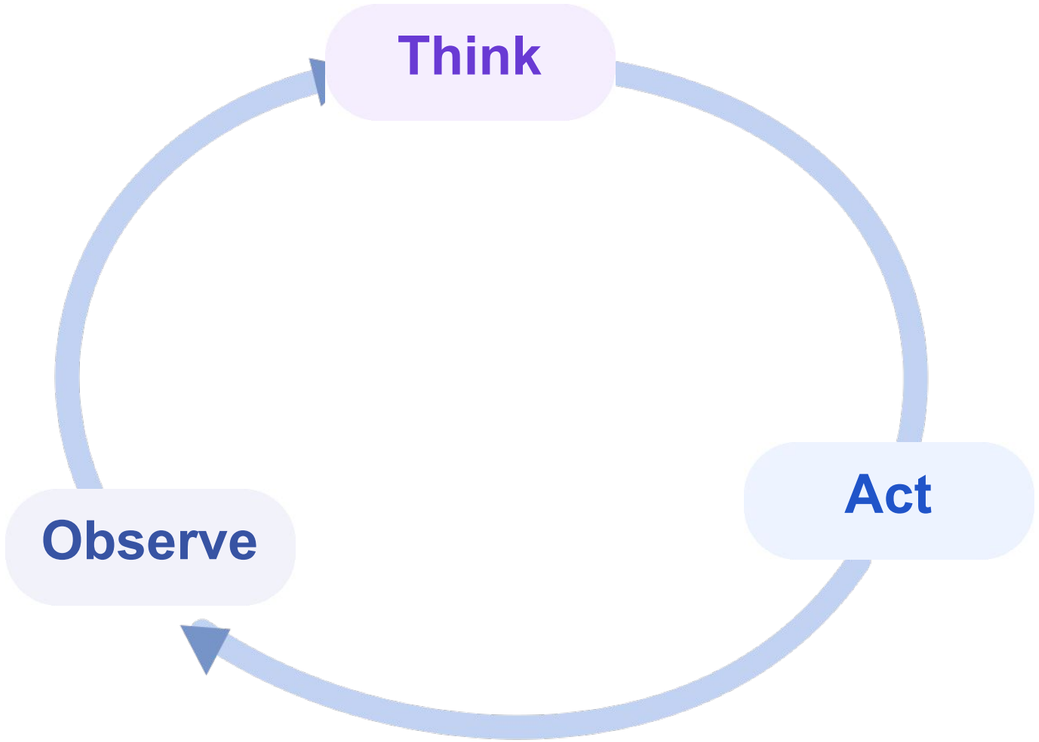
Book an advisor slot

LLM | Context | RAG | Tools | Agent

LLMOps across every layer

MCP joined the Linux Foundation’s Agentic AI Foundation in December 2025.

An agent is an LLM in a loop



“Help me plan next semester.”

Check my transcript.

Find open electives.

Flag timetable clashes.

Ask me to confirm.

Use the lowest rung that solves the problem

01

Single LLM call

Summarise

02

Workflow

Follow set steps

03

Router

Choose a path

04

Tool-using agent

Choose the steps

05

Multi-agent

Hand off work

Six shifts defining AI engineering in 2026

Ask which of these would change how you build the campus assistant.

Tool protocols

MCP exposes tools and data to the app.

Coding agents

Propose a patch, run tests and request review.

Voice interfaces

Stream speech, handle pauses and recover from interruptions.

Action controls

Check permissions and confirm consequential actions.

Local LLMs

Run a quantized model on a device or on-site server.

Fine-tuning

Train an adapter for a repeatable domain task.

AI fails politely, and LLMOps catches it

A confident wrong fee and a duplicate booking leave different traces.

01 EVALUATIONS

Catch a wrong fee

Pin 50 known questions. Score the fee, its source and when the assistant should abstain.

Release gate: compare with the baseline.

Trace the answer

Log retrieved page IDs, prompt version and tool result. Find the first step where the evidence went wrong.

Latency + cost

Track p95 latency, tokens and retries per successful task.

Action safety

Validate arguments. Confirm a booking and deduplicate retries.

Example incident review · A trace explains one request. An evaluation checks whether the fix generalizes.

What it takes to run an LLM locally

The model weights, runtime and application can all sit on the customer’s machine.



ON-SITE SERVER OR PERSONAL DEVICE

01 **Choose the weights**

Pick a compatible GGUF file and a quantization level.

02 **Start the runtime**

llama.cpp or Ollama serves the model locally.

03 **Call and measure**

Send a prompt. Record RAM, first-token time and tokens/s.

7B parameters × 4 bits ≈ 3.5 GB for raw weights. Budget extra memory for quantization metadata, KV cache and runtime.

How fine-tuning changes an answer

Train a repeatable behavior. Here, the campus assistant routes an unanswered question for review.

ONE LABELED TRAINING EXAMPLE

Student asks

“Is there a weekend lab fee?”

Desired output

```
{"action":"review","source":null}
```

LoRA / LOW-RANK ADAPTATION

$$W + \Delta W$$

Freeze the base weights W. Train two small matrices whose product supplies the update.

What to measure On unseen requests, does it choose the right action and return valid JSON?

From examples to a deployable adapter

Each step leaves a file another engineer can inspect or rerun.

01	**Curate**	Review prompt → response pairs. Split by source into train, validation and test.	**JSONL + Datasets**
02	**Train**	Load a pinned base and tokenizer. TRL SFTTrainer fits a PEFT LoRA adapter.	**PyTorch + TRL + PEFT**
03	**Evaluate**	Compare base vs. tuned on unseen cases. Track routing accuracy, schema validity and regressions.	**pytest + eval report**
04	**Package**	Version the base, adapter and chat template. Serve a supported adapter with vLLM.	**Model card + vLLM**




Hugging Face tooling and PyTorch

RAG or fine-tuning? Follow the failure.

RAG

FAILURE

The fee changed this term.

CHANGE

Update the fee sheet and index.

TEST

Did it retrieve and cite page 4?

Fine-tuning

FAILURE

It routes review cases poorly.

CHANGE

Train on reviewed examples.

TEST

Did held-out routing improve?

Use both

FAILURE

The desk needs fees in JSON.

CHANGE

Retrieve the fee, tune the format.

TEST

Check the value and the schema.

For fixed output contracts, try prompting, schema validation or constrained decoding before spending a training budget.

Where I’m betting on local models



M. HUZAIFA SHAHBAZ / LINKEDIN

25 May 2026

“Over the next few years, I expect we'll see a massive software shift back to "local-first" AI and perpetual-style licensing.”

My bet through 2028: a tuned small model can earn its place on a customer’s own hardware.

A perpetual license for a local AI product

Example offer: a campus triage assistant installed on one admissions-office server.

What ships

A versioned app, compatible base weights and a tuned routing adapter. Include the eval report and installer.

What the buyer gets

A one-time right to use release 1.0 on the agreed installation. The contract defines duration and scope.

What recurs

Hardware, power and maintenance. New adapters, support and major upgrades can be priced separately.

Before packaging Check model redistribution terms, notices and rights to training data.

Pick one of five lanes



AI / GenAI Engineer

LLM products

Agentic AI / Automation Engineer

Tools and flows

LLMOps / AI Platform Engineer

Reliability

Forward Deployed / AI Solutions Engineer

Client solutions

AI Product Builder / Freelancer

Paid pilots

A practical AI engineering stack

Pick one tool per job. Learn how data and errors move between them.




Service

Python · Git · FastAPI

A tested API, SQL migrations and a repeatable build.




Model

PyTorch · Transformers · PEFT

A model benchmark and a versioned adapter.




Delivery

PostgreSQL · Docker

A retrieval index, a deployed service and release checks.

Add tool calling and MCP when a task needs actions. Add tracing and evaluation before that task reaches a user.

Six months to engineer. Six months to prove it.

For students with basic Python. Plan for 12–15 focused hours each week.

MONTHS 01–06

Engineer a system

Build the service, retrieval pipeline, local model, adapter and tool workflow for one campus task.

Checkpoint: can a peer deploy it and debug a failure?

MONTHS 07–12

Build a portfolio

Take two products through a real pilot. Publish the trade-offs, evaluations and changes users drove.

Checkpoint: can a reviewer verify your claims?

Months 1–3 · Build the foundations

Start with a service another person can run. End with retrieval you can measure.

01 **Ship a backend**

MONTH 01

Build a FastAPI + PostgreSQL service with auth, tests and CI. Deliver a Docker image, deploy URL and one-command setup.

02 **Learn the model**

MONTH 02

Train a tiny causal model in PyTorch. Inspect next-token loss. Benchmark two inference models on the same 50 prompts.

03 **Engineer retrieval**

MONTH 03

Compare BM25, dense and hybrid search on labeled questions. Publish page hit@5, citation accuracy and three failure cases.

Quarter checkpoint • A classmate can reproduce your baseline and trace an incorrect answer.

Months 4–6 · Own a production-style system

Make choices about memory, model behavior and actions. Defend each one with evidence.

04 **Run inference locally**

MONTH 04

Test two quantizations on the same hardware and context size. Report quality, peak RAM, first-token latency and tokens/s.

05 **Tune one behavior**

MONTH 05

Curate labeled cases, hold out whole sources and train a LoRA. Compare it with a strong prompt baseline. Log regressions.

06 **Operate an agent**

MONTH 06

Deploy a tool workflow with auth, confirmations and safe retries. Add traces, eval gates, an alert and a tested rollback.

Month-six test • Rebuild from scratch, replay a timeout, show one valid booking and roll back.

Months 7–9 · Earn evidence from real users

Pick two products with different constraints. Both need a baseline and a named user.

07 **Find a pilot partner**

MONTH 07

Interview five users. Observe one task and time its baseline.
Agree on the data, owner, success measure and pilot scope.

08 **Pilot product A**

MONTH 08

Put a cited search assistant in front of those users.
Publish a redacted eval set, architecture and error analysis.

09 **Build product B**

MONTH 09

Ship a local-first triage tool for a second repeatable task.
Publish base vs. tuned quality and a hardware benchmark.

Quarter checkpoint • Two case studies have different engineering decisions a reviewer can inspect.

Months 10–12 · Make the portfolio inspectable

Ship improvements that follow from evidence. Make both products easy to try or replay.

10 **Harden a real action**

MONTH 10

Extend one product with a tool call and an approval step.
Publish a replay for a timeout, duplicate request and recovery.

11 **Run a user pilot**

MONTH 11

Observe three users completing tasks. Record where they stall.
Fix the largest failure. Repeat the test and show the change.

12 **Package your proof**

MONTH 12

Release both repos or redacted versions, plus 90-second demos.
Publish two case studies. Use one in each targeted application.

Graduation test • A reviewer can run it, inspect the numbers and see how feedback changed it.

Proof beats certificates

Put these four pieces of evidence on the first screen of your project.

EXAMPLE REPOSITORY

campus-assistant

A fee answer you can verify.

- README.md
- evals/report.csv
- traces/failed-request.json
- demo/90-second-walkthrough

01 Try it

Working release or short recorded walkthrough.

02 Rebuild it

Architecture, setup and pinned dependencies.

03 Measure it

Baseline vs. release, sample size, quality and latency.

04 Explain a failure

Trace → root cause → fix. Link the regression test.

A hiring manager can skim the story. An engineer can follow the evidence.

Where to find your first pilot in Pakistan



Pakistan

University admissions

Find a fee, scholarship rule or deadline with a cited source.

A local distributor

Route Urdu and English order queries for staff review.

Student society

Book a room, request approval and prevent double booking.

National AI Policy 2025 targets · Train 1 million people by 2030 · Offer 20,000 stipend-based internships annually

Day 1 starts Monday.

Tag me.

Open discussion and career Q&A

mhuzaifa.com/duet-genai

Connect on LinkedIn

/in/mhuzaifadev