HSF LLMs for HEP

Hackathon - Sep 28, 2026

Introduction

Plan for today

- Discuss the survey results, feedback from the community (15 min)
- Prioritize the topics, translate into chapters (5-10 min)
- Match names to topics and develop material (rest of the day)
- Open a PR, get recognition and eternal glory 🏋️🏆

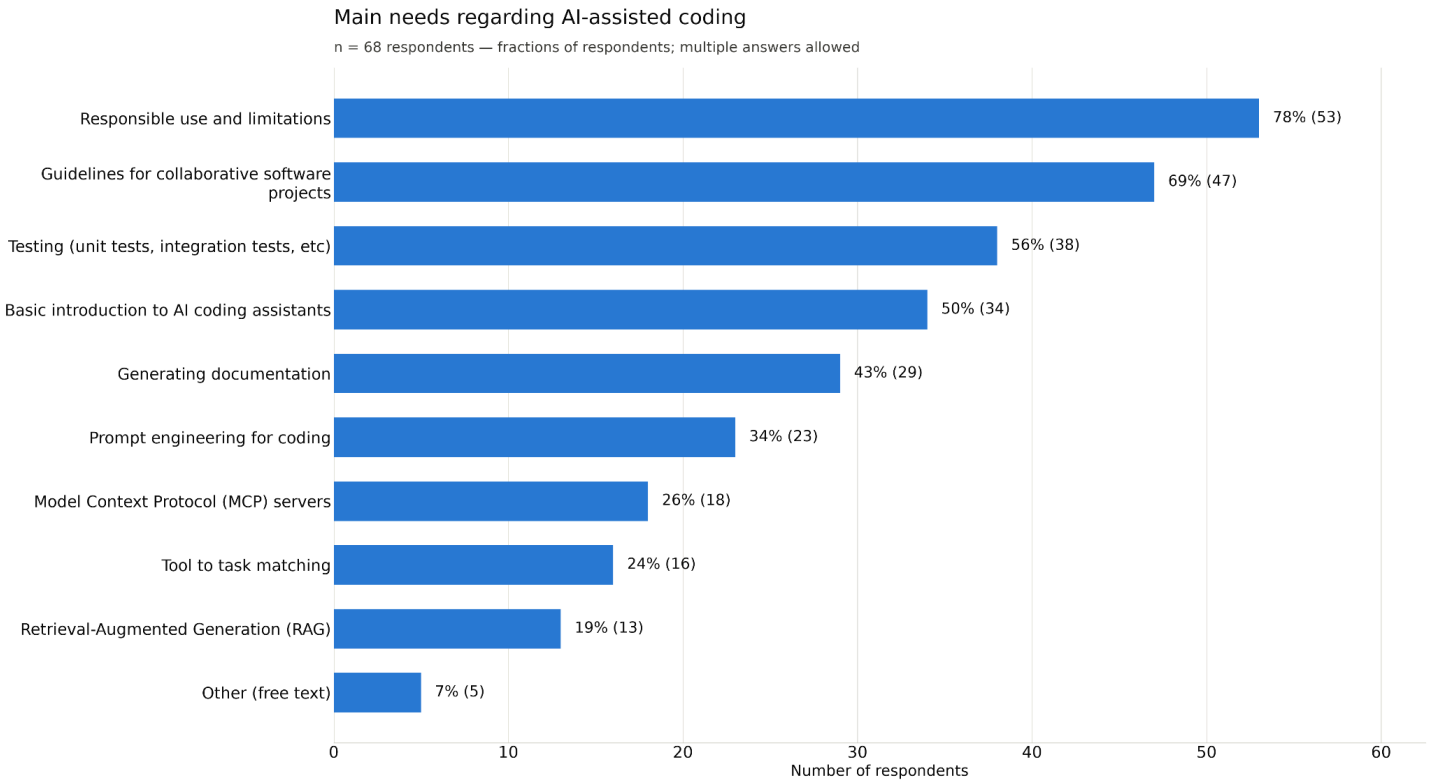
Repository

https://github.com/hsf-training/hsf-training-llms-for-hep

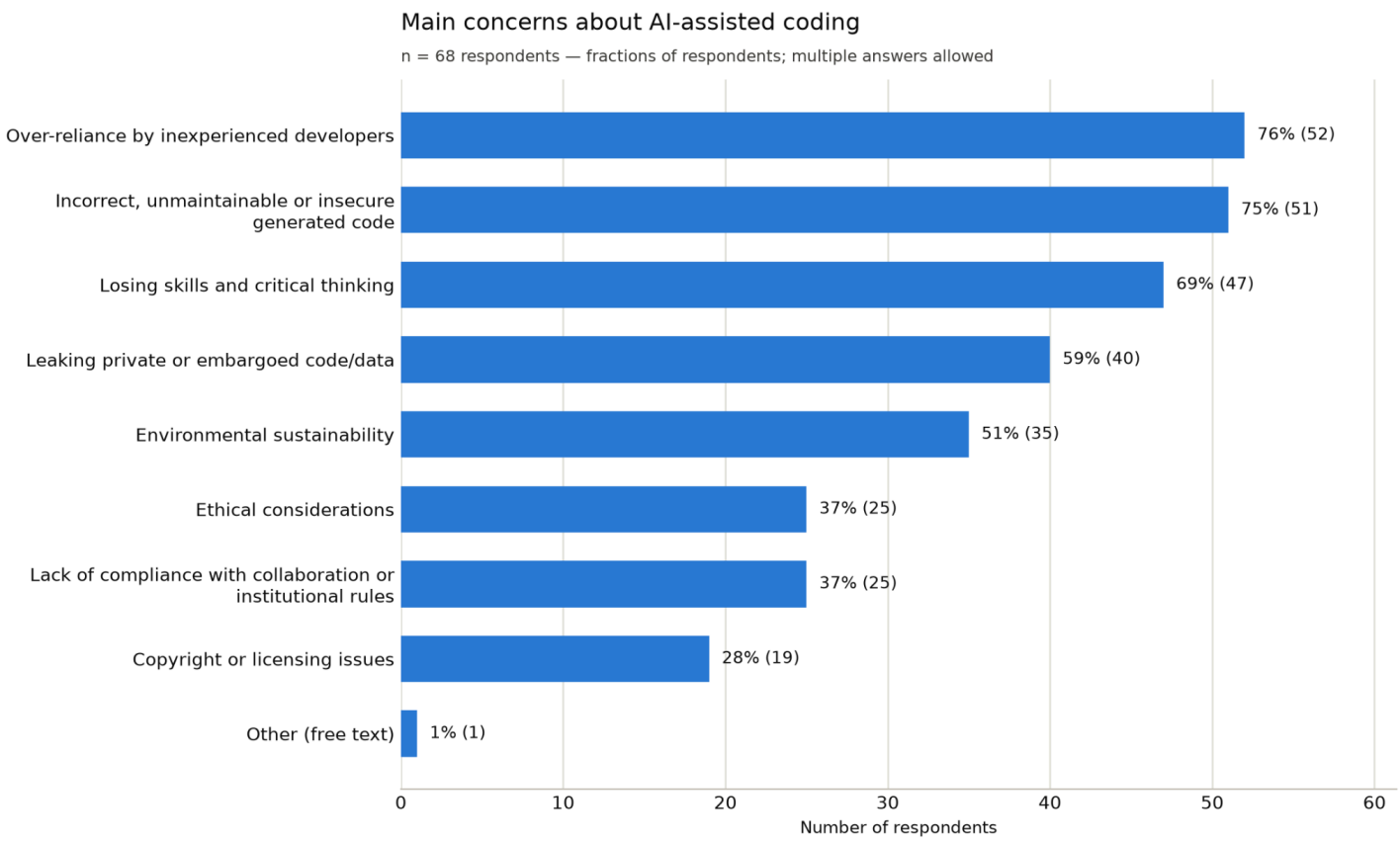
State of the survey

Training on effective AI-assisted coding in HEP and NP - for Individuals (Responses)

Main needs by the community



Main concerns



Use cases in HEP/NP considered particularity useful

- Assisting across nearly all code-related tasks, general
    - Plotting is frequently mention
- Recording the meeting + generating summaries
- Generation of documentation for code that is currently not maintained, usually because the main developer left the collaboration / academia
- Code review and commenting

HEP or Nuclear Physics specific examples that would make good training demonstrations

- Generic demonstrations using Python, Awkward, and ROOT
- Validation is pretty much emphasized
- How to maintain good guardrails
    - Unit, integration and functional tests as harness for LLMs
- Demonstrate the importance of proper curation and archival of analysis information for reproducibility and reusability
- Open Data resources

- Plotting (again)

What level of training would you consider useful?

1. Practical hands-on tutorial (75%)
2. Advanced usage on HEP workflows (71%)
3. Policy and responsible use (42%)
4. Introductory/awareness level (37%)

Current limitations

- Budget limitations, institutional licences not available (83%)
- Privacy concerns (57%)
- Lack of trust on AI-generated code (44%)

Main practices you would like to see emphasized

- Emphasize that humans remain fully responsible for AI-generated code
- Maintaining transparency and clear policies on copyrights, privacy, and code ownership
- Focus on safe and reliable usage through the setup of secure sandbox environments
- Effective prompt engineering to minimize token usage and environmental impact
- Proper documentation practices to help AI assistants digest code and handle codebases

Material already available (training, guidance)

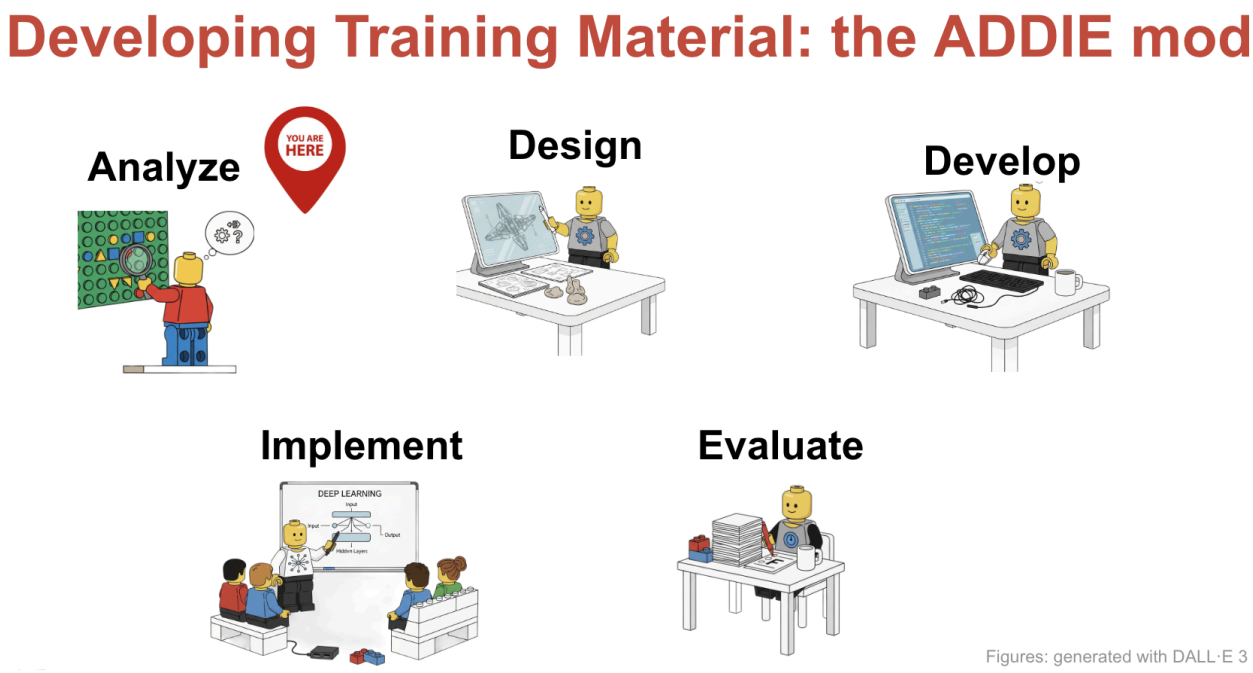
- AI coding in ATHENA, guidelines
- EIC AI-coding tutorial
- Skills for Mu2e
- LLMs and Prompt Engineering at @CompHEP 2026
- Agentic AI @ CompHEP 2026
- Responsible AI-assisted coding and common training aspects
    - Responsible AI Community Discussion
    - Reading material
- https://everse.software/RSQKit/ai
- https://aiforscience.eu/short-version/
- ...
- (Add more if you are aware of)

Misc

- “The key to using agentic AI efficiently seems to me to have a well curated and maintained set of skills and MCP servers”
- “Effective agent use requires a mixture of design skills, management skills, and physics intuition”
- “Collaborations must obey rules of many different institutions/countries at the same time”
- “Raising cost, AI adaptation for use cases where deterministic solution are would be better”

Discussion

- Our goal for today should be having a structure for a training module on “LLMs for HEP and Nuclear Physics”
    - It should be considered as a starting point, as the module will evolve as we gain a better understanding on “best practices”; the tools evolve as well
- Let’s avoid reinventing what is already available



Figures: generated with DALL·E 3

Analyze Training needs

Step 0: Who is the target?

- HEP and Nuclear Physics (NP) community
    - Newcomers that join experiments in a world dominated by AI-assisted coding

Step 1: let's give some thought to *why* we want to develop a training program

- We are surrounded by AI in every sphere and it is tempting to let the AI tool to drive
    - We consider **human must stay as the driver**
    - Scientific software requires precision - we can't simply trust that the LLM output is correct
- HSF is in a position to spread the word about the **best practices** (once we understand what are those)
    - In particular with newcomers
    - HSF responsible AI forum
- Teach what a good **prompt** is - the quality of the output will depend on it
- Stress that coding fundamentals are not optional
- Consider the resource utilization and **guardrails** when using agentic tools
    - Resource estimation
- Safety! Safety practices
    - **Sandboxing**
- Testing
- Proper handling of PII
- Documentation
- Tool: **Skills, MCP, RAG, Inspection**

Prerequisites (what the student should know before the training)

"Training is a solution that can fix gaps in knowledge and skills"

- Scripting (bash)
- Python, Scikit-HEP
- Containers (for sandboxing)
- Git

Required skills (what she/he should know after the training)

- Basic understanding on how LLMs work
- Prompt engineering
- Best practices:
    - Modular approach to develop
    - Testing: unit testing
- Safety practices
    - Sandboxing

    - Understanding of guardrails
- Toolkit: Skills
- Pipelines to automatize documentation

More advanced topics:

- MCP, basic understanding of APIs
- RAG (?)

Goals for a HSF training module

SMART objectives

- Specific: clearly state what the learner will be able to do
- Measurable: the outcome must be observable and quantifiable
- Achievable: should be realistic given the learner's background and the time available
- Result-oriented: focus on the outcome
- Time-bound: specify a timeframe for achieving the objective ("by the end of this module")

Our learning objectives

- By the end of the module, the trainee should have a basic understanding on how LLMs work, what are the best practices when using them, and how to use them safely
- Understand what are the limitations
- How to evaluate and validate what is the output from the LLM
- Be aware of common pitfalls on the usage of Agentic AI in science
- Understand your responsibility as user of AI tools

The HSF training material

The HSF-Training Design Philosophy emphasizes student-centric, hands-on, experiment-agnostic, reusable, and open content.

“Problem-centered, rather than content-oriented” (The adult learning theory)

Title

Probably we will have a better idea once we define the topics. Do we want to add “Responsible” on the title?

Topics/Chapters

- Introduction: basic understanding on how LLMs work [Quinn, Richa]
    - Responsible usage
        - Pairing of task-tool (do you really gain something using AI?)
    - Limitations
    - Code ownership
- Safety
    - Discussion on the different levels of sandboxing
    - Using containers [Alex]
    - Bubblewrap [Andres]
    - ...
- Prompt engineering [Iliomar]
- Toolkit:
    - AGENTS.md [Aashirvad]
    - Skills [Aashirvad]
    - memory [Aashirvad]
- Building Sustainable Software
    - Guidelines for collaborative projects, AI policies [Michel]
    - Making it modular, easy to test [Michel]
    - Mantainable: variable names, docstrings [Michel]
- Testing: pytest [Karan]
    - Understanding of guardrails
- Validation
    - pyflake, static code checkers [Leonid]
- Pipelines to automatize documentation [Karan, Alex]
    - Users, development
    - Integration with Non-AI tools (Sphinx)
- More advanced topics:
    - MCP, basic understanding of APIs [Arghya]
    - RAG [ ]
    - Remote execution [Leonid]

Setup

Minimize friction: pre-configured environments reduce time spent on setup, keeping focus on the actual learning (containers, GitHub Codespaces)

- Bring your own LLM
- Free open source models can be used [Andres]

    - Opencode
    - Ollama cloud (a bit of free usage)
    - Nvidia?
    - Codex free tier

Examples and exercises

Must be relevant and ideally have an immediate impact in the work of the trainee. Focus on (real!) problems in the HEP & Nuclear Physics context.

-