PFE Projects Book

Email us at contact@redxt.com with your CV and a letter of motivation about the project you are interested in and we will get back to you.

Academic Year 2025/2026

ReDX

Project A: Open-Source HPC Networking Chatbot (FabricsGPT)

1. Context and Motivation

In High-Performance Computing (HPC), the network fabric is as critical as compute and memory for scaling performance, ensuring predictable latency, and maintaining energy efficiency and total cost of ownership (TCO). Modern HPC clusters increasingly rely on advanced interconnect technologies such as InfiniBand (HDR, NDR, XDR) and high-speed Ethernet (100/200/400/800 GbE with RoCEv2 and DCB) to support massive AI and simulation workloads. These fabrics involve complex parameters—link speeds, ASIC generations, routing algorithms, congestion management (PFC, ECN), and offload capabilities (SHARP, GPUDirect, RDMA)—that collectively determine cluster performance and reliability. However, knowledge about these networking technologies is scattered across vendor documentation (NVIDIA/Mellanox, Intel, Broadcom), standards bodies (IEEE, OpenFabrics Alliance), whitepapers, HPC site guides, and community forums with best practices in architecting HPC networks for large scale clusters. These sources change frequently in layout, terminology, and available data fields. Integrating consistent and up-to-date networking knowledge into ReDX's **Cluster Configurator** or other HPC design tools is therefore time-consuming and fragile if done manually and requires substantial expertise and experience.

Goal: Build an autonomous, domain-specialized, LLM-assisted networking design assistant (**GPTFabrics**) that continuously discovers, interprets, normalizes, and reasons over InfiniBand and Ethernet information—and exposes valid insights (topology design, bandwidth/latency estimation, congestion control, offload compatibility) to the Cluster Configurator.

2. Problem Statement

Build a system that can, to varying degrees and by any justified approach:

- **Discover** relevant networking information (InfiniBand, Ethernet, topologies, RDMA/RoCE support, congestion control) from heterogeneous sources without relying on fixed structures or vendor-specific assumptions.
- **Understand** what it finds by normalizing networking concepts such as link speed (Gb/s), port radix, ASIC generation, latency, routing scheme …
- **Support design reasoning** (e.g., "Is this 2-tier fat-tree non-blocking for 128 GPU nodes?", "What's the oversubscription ratio with 200 Gb/s downlinks and 400 Gb/s uplinks?")
- **Remain robust over time** despite evolving nomenclature, topologies, and hardware generations (HDR→NDR→XDR, 400→800 GbE).

3. Tools and Resources:

- **Hardware:** 2–8 × NVIDIA H100 GPUs, Lustre filesystem, HPC Cluster
- **Software:** HuggingFace Transformers, NVIDIA NeMo, FAISS, LangChain or LlamaIndex
- **Data Sources:** NVIDIA/Mellanox docs, Intel and Broadcom Ethernet specs, IEEE papers, OpenFabrics Alliance resources, academic and industrial HPC network case studies.

4.Learning Objectives

The intern will work in **close collaboration and mentorship with a PhD candidate pursuing world class publishable research** within an industrial and practical environment.

By the end of the internship, the student should be able to:

- Analyze the **HPC fabric landscape**, differentiating InfiniBand and Ethernet in terms of latency, throughput, congestion behavior, and scalability.
- Design **data and retrieval pipelines** resilient to heterogeneous and evolving documentation formats.
- Compare **alternative reasoning strategies**: rule-based bandwidth calculators, semantic search, LLM prompting, fine-tuning, and hybrid RAG pipelines.
- Define **evaluation metrics** (precision/recall for field extraction, correctness of topology reasoning, hallucination rate, adaptability to new generations).
- Communicate clearly the **design trade-offs, limitations, and interpretability** of the system

5. System Capabilities

- **Find:**Locate pages, datasheets, or topology design guides that describe InfiniBand and Ethernet hardware (NICs, switches, optics, cables), congestion management techniques, and topology templates.
- **Extract/Map:**Identify and normalize key fields: link speed (Gb/s), port count, ASIC generation, latency, buffer depth, PFC/ECN support, RDMA/RoCE capabilities, routing type, topology constraints, and oversubscription ratios.
- **Explain:**Generate concise explanations for networking concepts such as “lossless Ethernet,” “non-blocking fat-tree,” “PFC vs ECN,” “RDMA over Converged Ethernet,” or “SHARP in-network reduction.”
- **Estimate:** Perform topology-aware calculations (node scalability, bisection bandwidth, port utilization, oversubscription analysis) and flag likely configuration issues (e.g., asymmetric leaf/spine counts or mixed link speeds).
- **Track Change:** Detect new hardware generations or unfamiliar field names and flag them for human inspection instead of silent failures—ensuring long-term maintainability as technologies evolve.

The student may implement only a subset of these features deeply, provided the methodology and evaluation remain rigorous.

6. Required Skills

The intern is expected to have high level understanding of these concepts and capable to learn a variety of required skills for the project including but not limited to:

- **Python programming:** HTTP requests, HTML parsing, data processing.
- **Networking fundamentals:** InfiniBand, Ethernet, RDMA/RoCE, PFC/ECN, topology design (fat-tree, Dragonfly, leaf-spine).
- **Data normalization:** units, naming standards, schema definition (speed, radix, latency, congestion control).
- **Database management:** DuckDB / PostgreSQL for structured data storage and retrieval.
- **Documentation skills:** clear technical writing and result presentation.
- **Intro AI/LLM skills (preferred):** prompting models to interpret technical text, map unstructured fields into schemas, and produce factual summaries of HPC network concepts.

7. Duration and Other Details

- **Recommended Period:** 6 months
- **Compensation:** Monthly stipend with potential end-of-internship performance bonus and potential paper publication co-authorship

ReDX

Project B: Dynamic RAG Dataset for Multi-Cloud HPC

Description

This internship focuses on teaching our model to think like a cloud architect for HPC. Today, our LLM understands HPC code, but it can't yet turn that code into a concrete hosting plan on AWS, GCP, or Azure—let alone a mixed, multi-cloud setup. The aim is to collect full, functional architecture examples (single-cloud and heterogeneous multi-cloud), capture all the moving parts (compute families, storage tiers, networks, schedulers, quotas/limits, regions) and their costs, and shape them into a clean, provider-agnostic schema. Alongside that dataset, we want to build a weekly auto-refreshed knowledge base that keeps SKUs, prices, and capabilities up to date so the model's answers are grounded in facts, not guesses.

Finally, there will be a fine-tuning stage of the LLM on that curated dataset so it can read an HPC repo profile, weigh trade-offs, recommend an end-to-end cloud architecture, and—when helpful—emit a diagram that makes the design easy to understand.

Current status:

The current status of the existing LLM is fine-tuned on HPC code repositories (it understands code), but it does not know clouds and cannot map HPC → cloud architecture yet.
The intern will work in close collaboration and mentorship with a PhD candidate pursuing world class publishable research within an industrial and practical environment.

PFE project Goals:

1. **Collect & Curate Deployable Architectures**
    - Build a dataset of **single-cloud and multi-cloud (heterogeneous)** examples across **AWS, GCP, Azure**; fill the common schema, add short rationales and **cost snapshots**, and (optionally) a small diagram per example.
2. **Provider Component Catalogs**
    - For each provider, collect all relevant components (compute, storage, network, scheduler) with the necessary specs, limits, regional availability, and pricing.
3. **Hint Parsing & Mapping to Cloud**
    - Parse the user hint (desired hardware/architecture/budget), validate it against the given codebase, and map requirements to best-fit cloud components per layer (compute, storage, network, HPC cluster choice) and rank providers for each part.
4. **End-to-End Architecture Synthesis (with RAG)**

   - Have the LLM assemble a complete hosting architecture for the event, grounded by RAG (retrieved provider facts), producing a clear, structured plan (and optional diagram).
5. **Auto-Refresh Mechanism for RAG**
   - Implement a scheduled refresh (e.g., weekly) to update SKUs, prices, limits, and docs across providers, keeping recommendations current and factual.
6. **Evaluation & Demonstration**
   - Evaluate on unseen repos for fit-to-needs, clarity, and cost awareness; deliver a live demo where the LLM outputs an accurate, end-to-end, cloud-agnostic (or multi-cloud) architecture for a given HPC codebase.

Required skills:

- **ML/NLP basics:** dataset design, prompt/response schemas, instruction fine-tuning.
- **Cloud literacy:** familiarity with AWS/GCP/Azure building blocks (instances/VMs, storage, regions, pricing).
- **Data tooling:** Python, JSON, simple ETL/versioning; basic vector search/RAG.
- **Good practices:** clear documentation, neat labeling, reproducibility with Git.
- **HPC basics:** Awareness of **GPU/CPU differences**, schedulers (K8s/Slurm/Batch), and why networking/storage matter for parallel jobs.
- **Software practices:** Git, code reviews, clear documentation, and an eye for user experience and safety (validations, guardrails).

Deliverables:

- **Cleaned, preprocessed, well-presented dataset** of **HPC cloud architectures** (both single-provider and **heterogeneous multi-cloud**), in our common schema, with short rationales and cost snapshots; (optional) diagram per example.
  Auto-refreshed RAG dataset (tunable periodicity) covering at least 3 providers with components, SKUs, limits, regional availability, and prices, plus a simple retrieval API.
  Fine-tuned LLM layers on your architecture dataset (config + checkpoints or LoRA adapters).
- **Final capability demo**: the LLM accurately recommends an end-to-end cloud hosting architecture for a given HPC codebase—detailed, cost-aware, fact-grounded, and (when useful) accompanied by a diagram.
- **Technical report**: schema, curation process, RAG refresh, fine-tuning setup, and evaluation results.

Duration and other details:

- Recommended period: 6 months.
- Compensation: Monthly stipend with potential end-of-internship performance bonus and potential paper publication co-authorship