Project proposal: Fundamentals of Cloud Computing, Fall 2026

- **Project Name:** Computational Storage
- **Project Proposer:** (Name, Email): Alex Merenstein (alex.merenstein@ibm.com), Vasily Tarasov (vtarasov@us.ibm.com), Anthony Hsu (Anthony.Hsu@ibm.com)
- **Open Source:** (Yes / No) Yes
- **Mentors:** (Name, Email): Alex Merenstein (alex.merenstein@ibm.com), Vasily Tarasov (vtarasov@us.ibm.com), Anthony Hsu (Anthony.Hsu@ibm.com)

Preferred Experience

- **Required (for most or all team members):** Linux command line familiarity
- **Required (at least one team member):** Python
- **Nice to have:** Familiarity with containerization, serverless

Project Background

Moving data into and out of storage systems for processing is often an expensive endeavor, either literally in dollars per byte or in time spent waiting on data movement. Hyperconverged architectures, which co-locate compute and storage resources, aim to address this problem by processing data where it resides. However, hyperconverged architectures have challenges in terms of scalability and management difficulties. Computational storage offers a different approach to combining processing and storage, which we propose to achieve by layering a compute layer on top of an existing storage system.

Project Description

The team will build two interfaces that expose “computation” at different levels of abstraction:

- A generic serverless-like interface that allows users to safely run arbitrary operations to process files.
- A data-conversion service that runs on the storage system that converts files into different formats. The conversion happens automatically in the background. The service exposes an API that lets applications indicate a preference for what format they would like data.

Additionally, the team will implement a baseline that processes data in a traditional manner (copying data from the storage system, processing it, then copying it back). This baseline will be used to evaluate the computational storage implementations in terms of both performance and quantity of data transferred.

Learning Outcomes

- Familiarity with foundational cloud native technologies including Kubernetes, containers (e.g., Docker), and serverless platforms
- Experience with advanced storage systems, e.g., Ceph
- Experience building distributed, cloud based systems
- Experience with common structured data formats such as Parquet