1 Team Plan and Work Distribution

This project was a collaborative effort between two members, focusing on a high-performance CUDA AutoEncoder. The workload was balanced between **System Architecture** and the **Computational Engine**.

1.1 Work Distribution

Member	Primary Roles	Key Contributions
Nguyen Long	System Architecture & GPU Optimization	Designed the core framework with smart memory allocation and shared memory intake to minimize overhead. Implemented Conv2D forward and backward passes including biases. Optimized performance using warp-level reductions and CUDA streams.
Nguyen Minh Nhat	Computational Engine & Evaluation	Implemented high-performance implicit im2col and GEMM-based convolution (Phase 2.4). Developed ReLU, Upsampling, and MaxPool kernels. Integrated the SVM training and evaluation pipeline.

Table 1: Summary of Team Responsibilities

1.2 Task Breakdown and Timeline

The development followed a four-phase approach, concluding with the final presentation on December 21, 2025.

- **Phase 1: Foundation & Research (Oct 31 – Nov 28)**
  - Initial project setup and CMake build configuration.
  - Implementation of baseline CPU convolution and image loading functionality.
- **Phase 2: Core Engine Development (Dec 3 – Dec 10)**
  - Created Tensor classes and established layer interaction protocols.
  - Developed the training loop, optimizers, and gradient clipping logic.
  - Implemented the backward pass for convolution and pooling layers.
- **Phase 3: Advanced Optimization (Dec 11 – Dec 18)**
  - Replaced standard reductions with warp-level reductions for tree operations.

  - Transitioned to implicit im2col for significant GPU speed gains.
  - Enhanced parameter handling and memory efficiency across the network.
- **Phase 4: Finalization & Evaluation (Dec 19 – Dec 21)**
  - Completed Phase 2.4 GPU implementation using GEMM-based convolution.
  - Finalized SVM integration for training feature evaluation.
  - Documentation, README completion, and Video Presentation preparation.

1.3 Contribution Percentage

Student Name	**Responsibility**	**Contribution %**
Nguyen Long	Architecture / Memory / Bias / Kernels	50%
Nguyen Minh Nhat	Math Engine / GEMM / SVM / Kernels	50%

Technical Note: *The architectural design utilizes shared memory to allow layers to exchange data efficiently without maintaining separate copies, while smart memory allocation ensures the system remains performant during intensive training epochs.*