Custom Convolutional Neural Network (CNN) Design and Benchmarking

Assignment Objective

This assignment focuses on the practical skill of designing a Convolutional Neural Network (CNN) from scratch. You will select a simple, manageable benchmark dataset, design a custom CNN architecture optimized for its specific image size and class count, and provide a full visual and code-based documentation of your design.

Task 1: Dataset Selection and Analysis

You must select **one** of the following simple benchmark datasets for your classification task:

Dataset Name	Image Count (Train/Test)	Image Resolution	Number of Classes	Challenge Focus
MNIST	60,000 / 10,000	28 \times 28 \times 1$ (Grayscale)	10 (Digits 0-9)	Handwritten Digit Classification
Fashion-MNIST	60,000 / 10,000	$28 \times 28 \times 1$ (Grayscale)	10 (Clothing items)	Simple Image Classification
CIFAR-10	50,000 / 10,000	$32 \times 32 \times 3$ (Color RGB)	10 (Objects like cars, dogs, planes)	Small Color Object Classification

1. **Dataset Choice:** State clearly which dataset you have chosen.
2. **Image Analysis:** Explain how the specific dimensions and color channels of your chosen dataset (e.g., $28 \times 28 \times 1$ or $32 \times 32 \times 3$) will influence your choice of initial kernel sizes, strides, and the number of pooling layers.

Task 2: Custom CNN (MiniNet) Architectural Design

Design a novel CNN architecture (call it "MiniNet") specifically optimized for your chosen dataset. The goal is to maximize efficiency and feature extraction for small-resolution images.

1. **Design Requirements:** Your MiniNet must have between **5 and 7 total layers**, organized into:
    - **Feature Extraction Block(s):** A sequence of at least two `Conv2D` layers followed by a `Pooling` layer.
    - **Classifier Head:** A `Flatten` layer followed by two or more `Dense/Fully Connected (FC)` layers.
    - **Activation Functions:** Use `ReLU` after all hidden layers and `Softmax` for the final output layer.
2. **Layer-by-Layer Specification:** Complete the following table for your final MiniNet design. This table acts as your architectural blueprint.

Task 3: Visual Representation and Code Structure

1. **Conceptual Diagram/Drawing:** Create a conceptual drawing that illustrates the flow of data through your MiniNet. Use simple boxes to represent the feature maps, showing how their dimensions shrink (H and W) after each pooling layer, and how the depth (C) increases after each convolutional layer
   .
2. **Python Implementation Plan:** Provide a well-commented Python code snippet defining your MiniNet architecture using a deep learning framework's structure (e.g., a Python class in PyTorch or a Keras Sequential model). This demonstrates how your blueprint translates directly into code, showing the exact parameters passed to each layer.

Deliverables

1. **CNN Design Report (This Document):** Contains Task 1 (Dataset Choice & Analysis) and the completed Layer-by-Layer Specification Table (Task 2).
2. **Architectural Drawing:** The conceptual diagram/drawing (Task 3).
3. **MiniNet Code Snippet (`MiniNet.py`):** The Python implementation plan for the architecture (Task 3).