PCA+CNN IMPLEMENTATION

PSEUDO CODE

```
1. Import necessary libraries for data handling, image processing, machine learning, and deep learning.

2. Mount Google Drive and define the path to the image dataset.

3. Define a function to:
   - Load images from paths
   - Resize them to 224x224
   - Extract corresponding labels
   - Return image and label arrays

4. List all image paths from the dataset folder.

5. Use the image loader function to load images and labels.

6. Convert class labels into binary format using label binarization.

7. Split the dataset into training and testing sets (90% training, 10% testing).

8. Flatten the image arrays and standardize the data.

9. Apply PCA to reduce image features to 20 principal components for both train and test data.

10. Define a function to:
```

- Create a fixed number of clients (e.g., 6)
- Randomly shard the training data across them
- Return a dictionary mapping clients to their data

11. Define a function to:
    - Convert each client’s data into TensorFlow datasets
    - Shuffle and batch the data

12. Create and batch data for each client and also for the test set.

13. Define model training parameters:
    - Number of communication rounds
    - Loss function, optimizer, and evaluation metrics

14. Define helper functions:
    - Compute client weight scaling factor based on data size
    - Scale model weights using a scalar
    - Aggregate scaled weights from all clients
    - Evaluate global model performance on the test set

15. Define a function to create a CNN model:
    - Reshape PCA input
    - Apply multiple Conv2D and MaxPooling layers
    - Flatten and add Dense layers
    - Output layer uses softmax activation

16. Initialize the global CNN model.

```
17. For each communication round:
    a. Distribute global model weights to clients.
    b. For each client:
       - Create local model and compile it.
       - Train model using client data.
       - Scale the trained weights.
       - Clear the session.
    c. Aggregate the scaled weights.
    d. Update global model with averaged weights.
    e. Test global model on test data.
    f. Collect evaluation metrics like accuracy, precision, recall, F1-score, sensitivity, specificity.

18. Plot performance metrics across communication rounds:
    - Accuracy
    - Precision
    - Recall
    - F1-Score
    - Sensitivity
    - Specificity
```