**ICP 6**

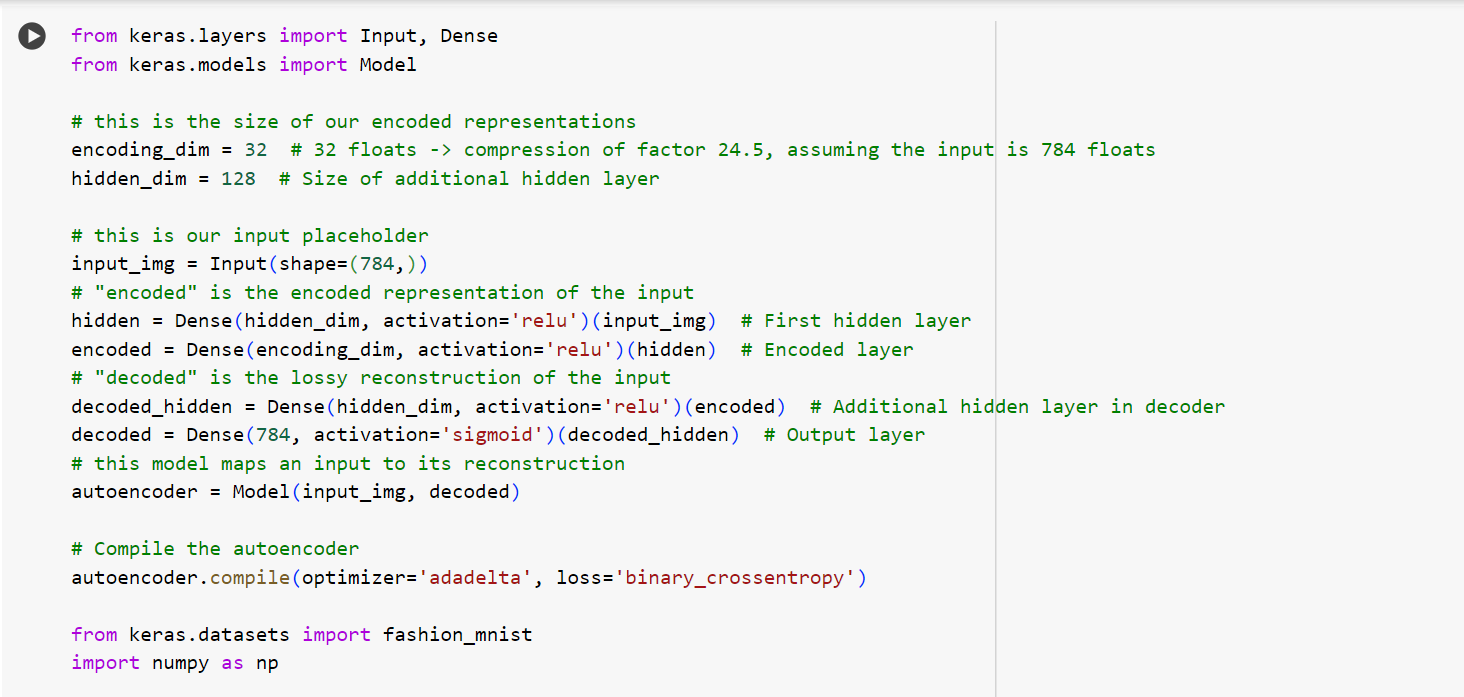
Kavya Reddy Nagulapally

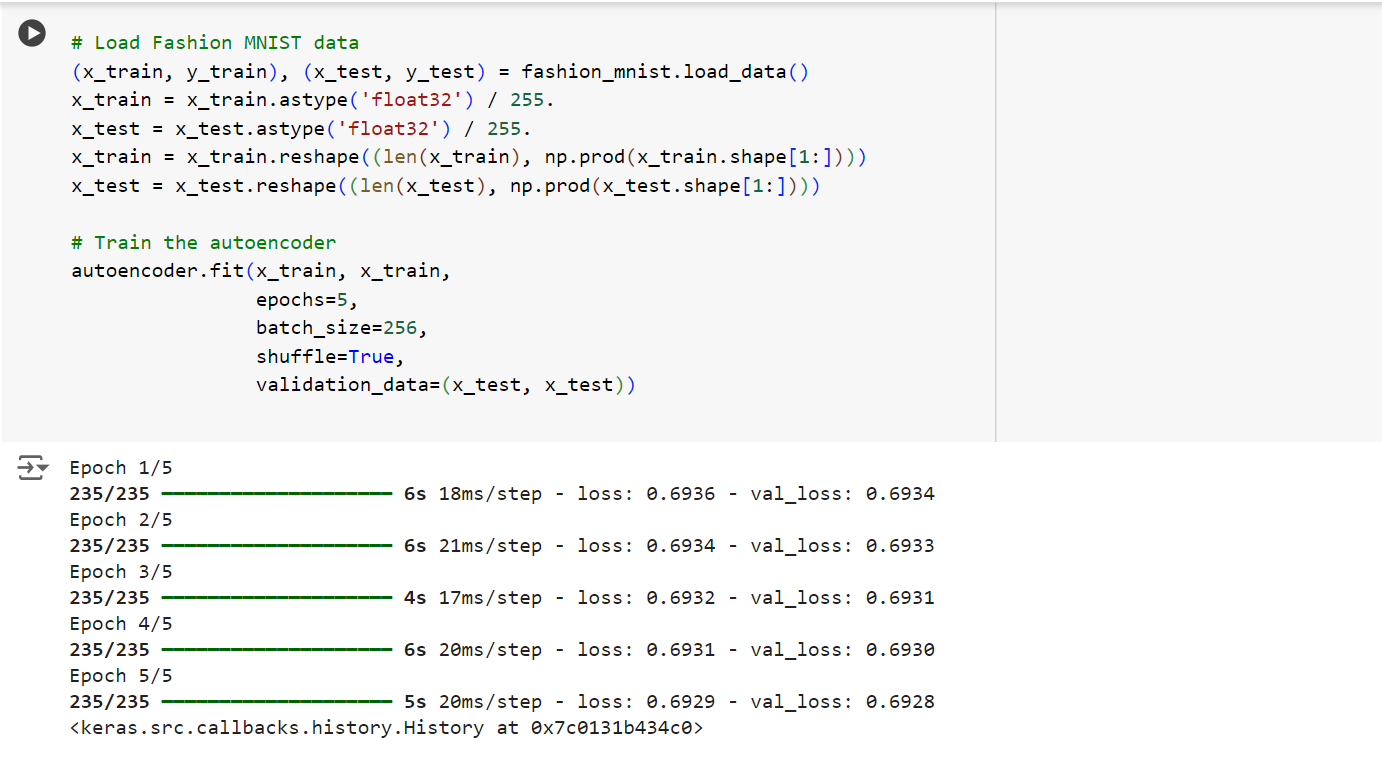
700759486

Github: <https://github.com/Kavyareddy03/ICP-6>

Question1

Code and Output:





**Explanation:**

This code builds and trains an autoencoder using Keras to compress and reconstruct images from the Fashion MNIST dataset. The input is a 784-dimensional vector (28x28 pixels), passed through a hidden layer of 128 neurons, then compressed to 32 dimensions (encoded representation). The decoder reconstructs the image by passing the encoded vector through a hidden layer and then outputting a 784-dimensional vector. The model is compiled with the Adadelta optimizer and binary cross-entropy loss. The data is normalized and reshaped, and the autoencoder is trained for 5 epochs on Fashion MNIST images.

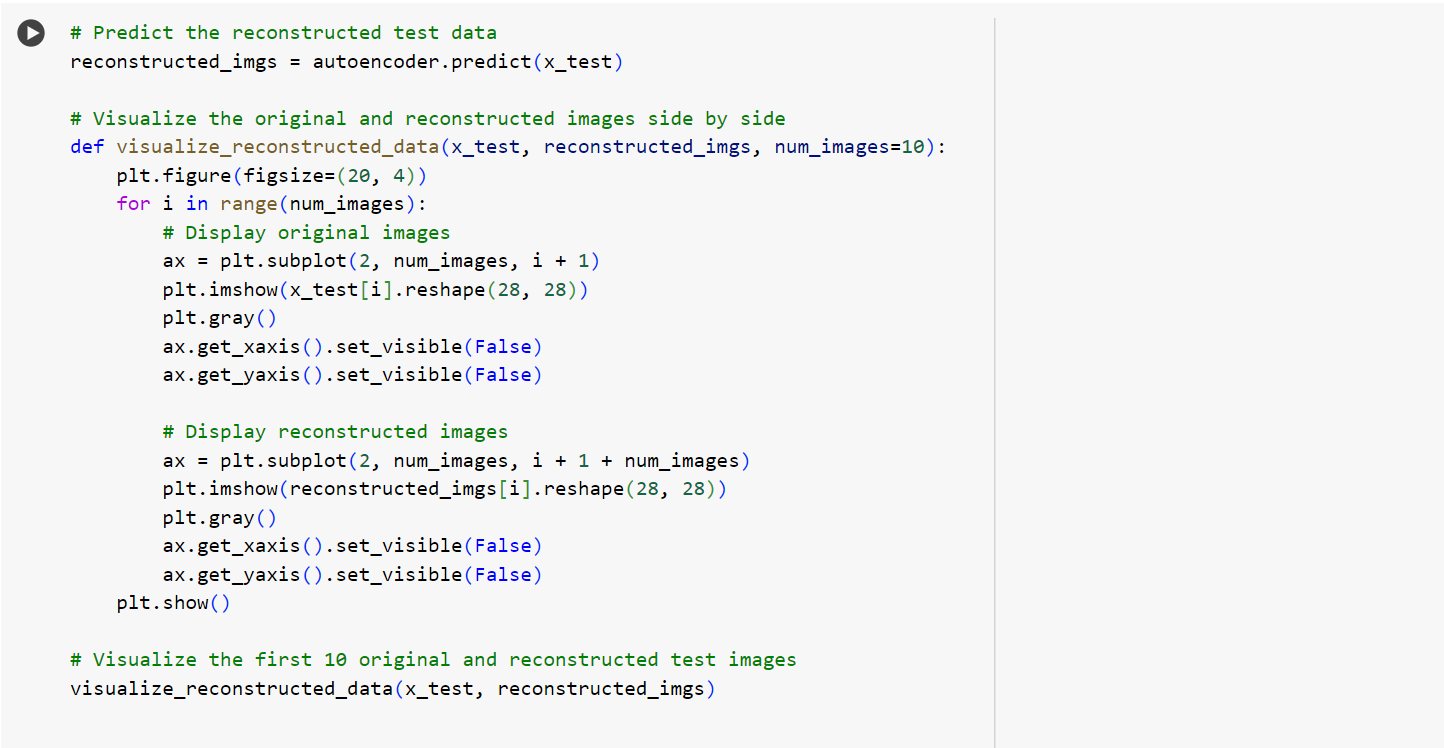
Question2

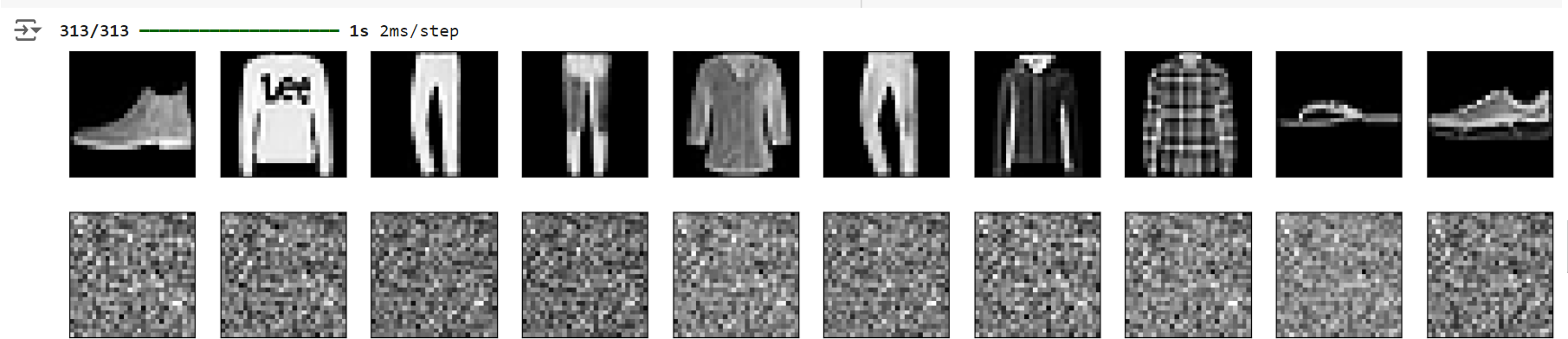
Code and Output:



**Explanation:**

This code visualizes the first 10 test images from the `x\_test` dataset using Matplotlib. The function `visualize\_original\_data` creates a figure and iterates over the specified number of images (default 10). Each image is reshaped from a flattened vector back to its 28x28 pixel form and displayed in grayscale. The axes are hidden for a cleaner presentation, and the images are shown in a row. The `visualize\_original\_data` function is then called to display the first 10 test images.





**Explanation:**

This code visualizes the original and reconstructed test images side by side using Matplotlib. First, the autoencoder predicts the reconstructed versions of the test images. The `visualize\_reconstructed\_data` function then displays the first 10 original images in the top row and their corresponding reconstructed images in the bottom row. Each image is reshaped back to its 28x28 pixel form and displayed in grayscale, with hidden axes for a cleaner look. Finally, the function is called to display both the original and reconstructed test images for comparison.

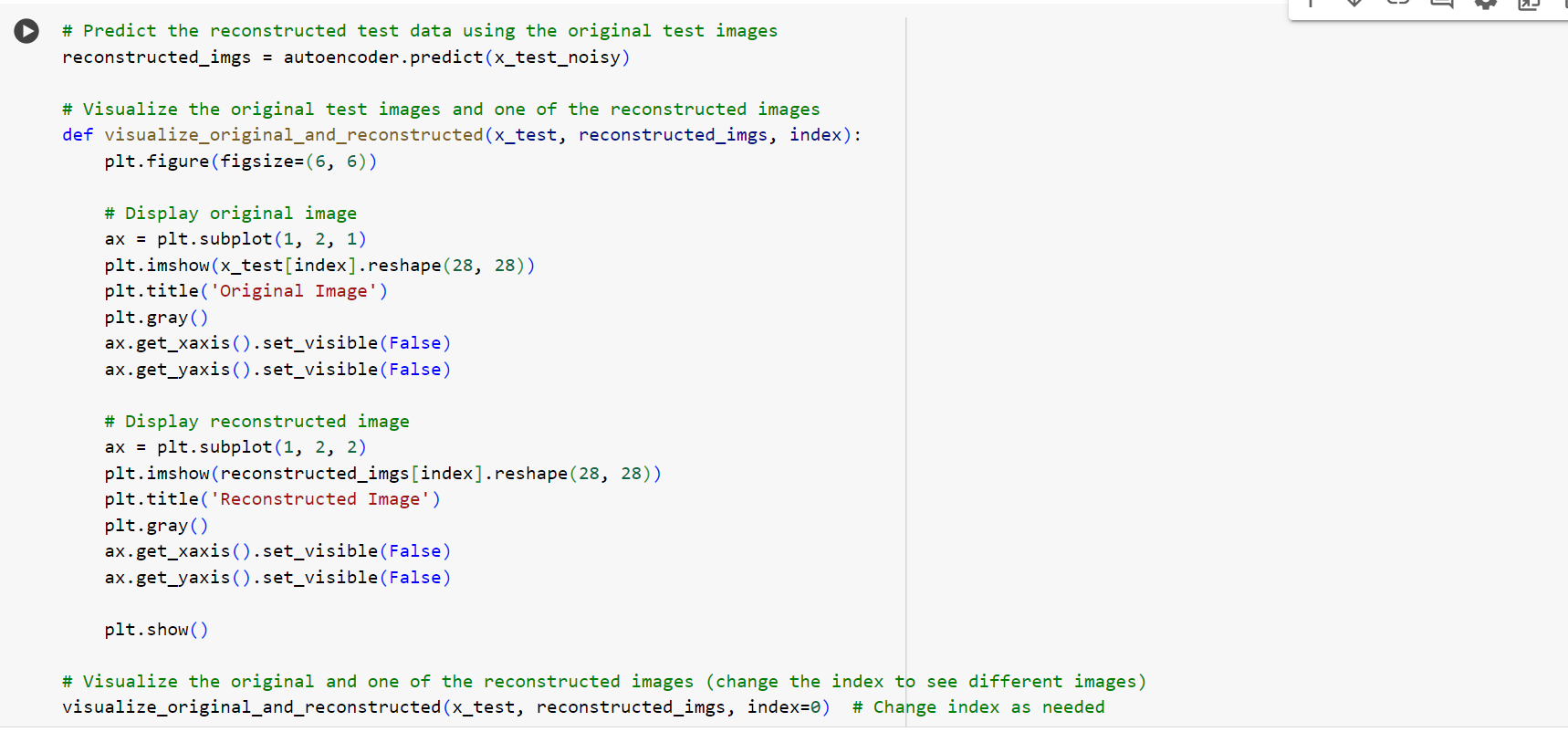
Question 3

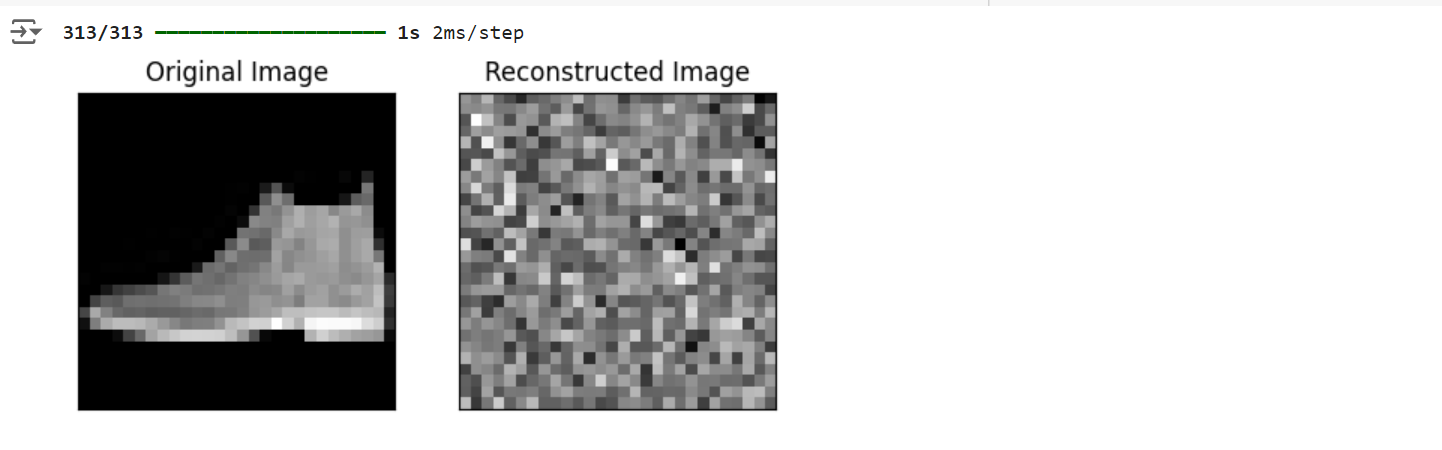
Code and Output:



**Explanation:**

This code visualizes noisy test images using Matplotlib. The function `visualize\_noisy\_data` takes a set of noisy images (`x\_test\_noisy`) and displays the first 10 images. It creates a figure and iterates over the specified number of images, reshaping each image from a flattened 28x28 format and displaying it in grayscale. The axes are hidden for a cleaner presentation, and the images are shown in a row. Finally, the function is called to display the first 10 noisy test images for visualization.





**Explanation:**

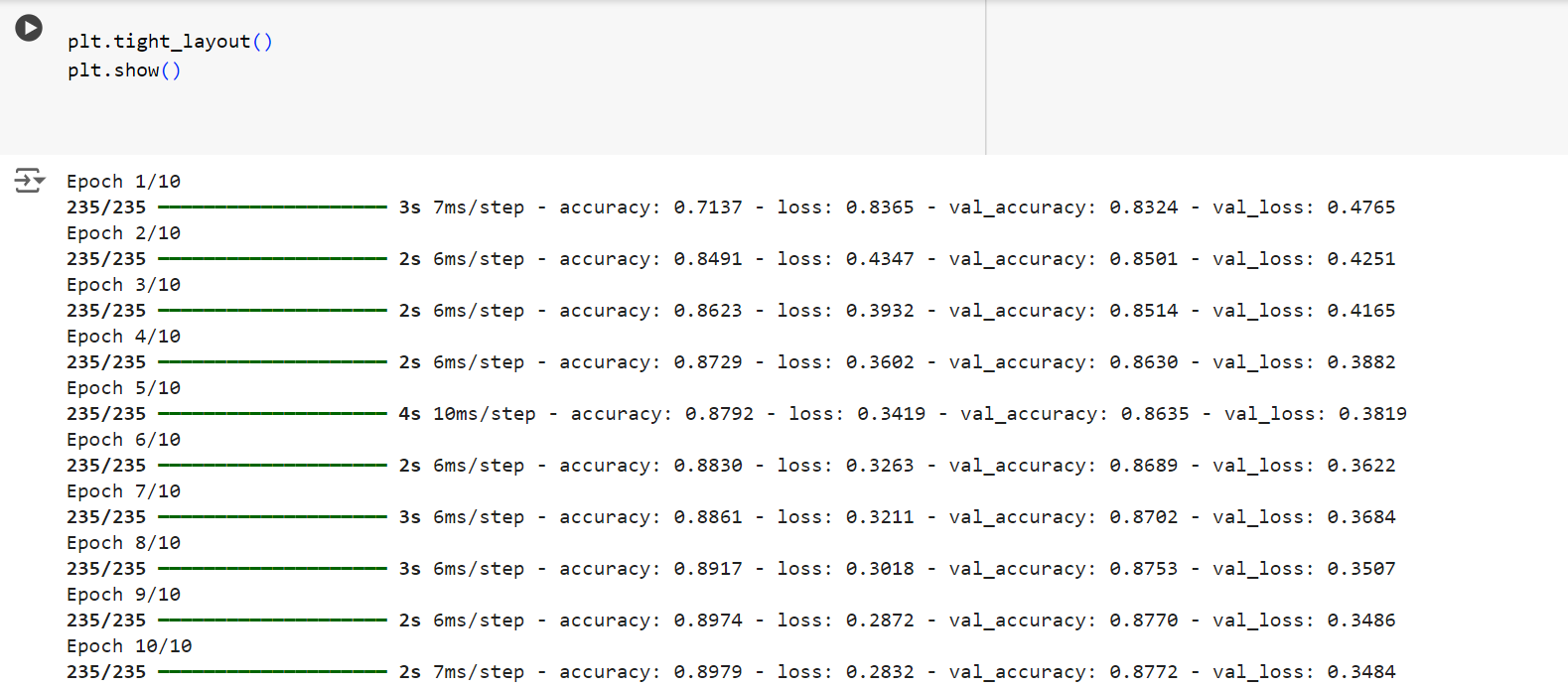
This code visualizes a side-by-side comparison of an original test image and its reconstructed version. The function `visualize\_original\_and\_reconstructed` takes the original test images (`x\_test`), the reconstructed images (`reconstructed\_imgs`), and an index as input. It plots the original image on the left and the corresponding reconstructed image on the right. Both images are reshaped to their 28x28 pixel form and displayed in grayscale with hidden axes for clarity. The function is called to show the original and reconstructed image at the specified index (index 0 in this case).

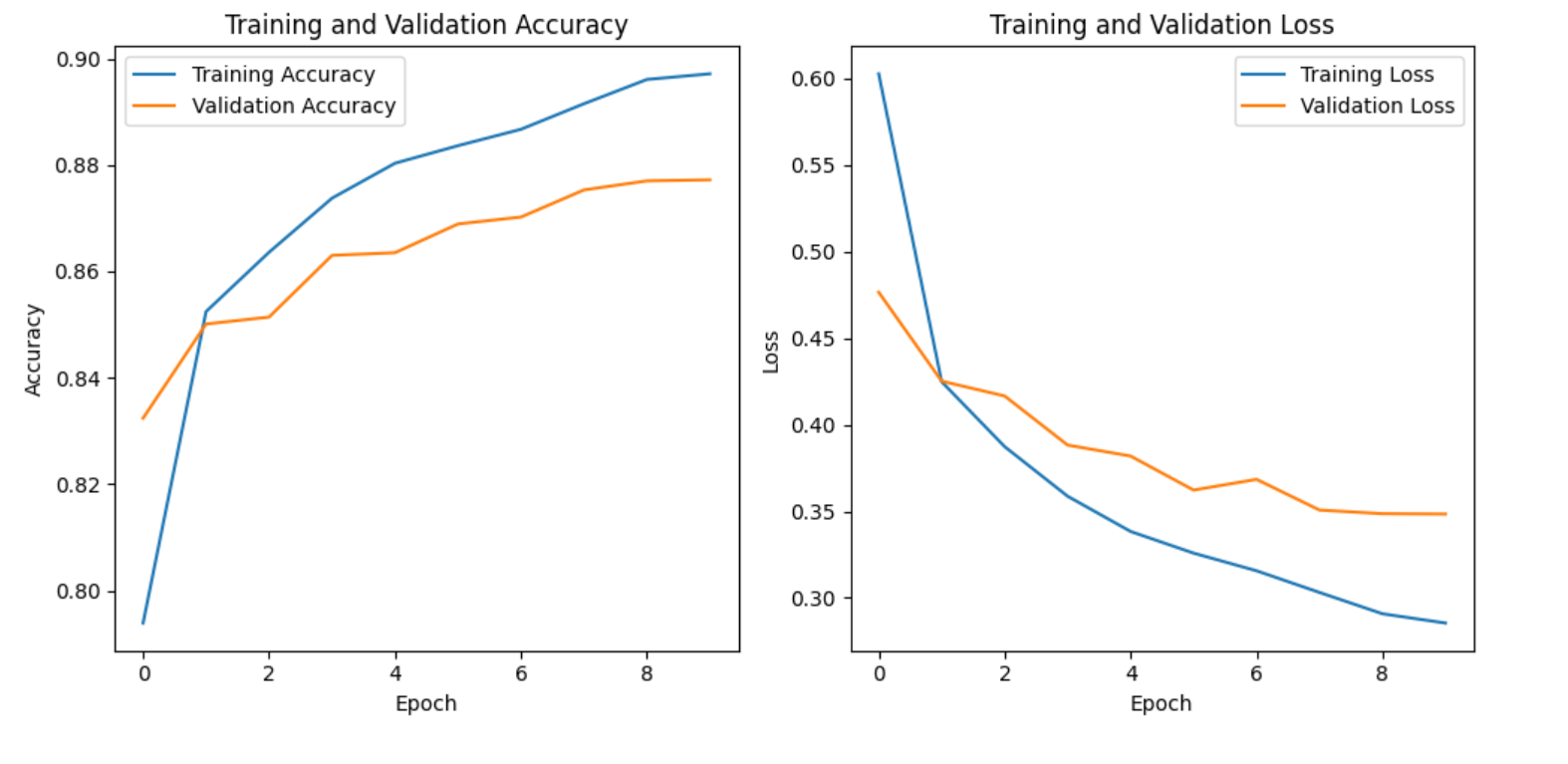
Question 4

Code and Output:









**Explanation:**

This code builds a neural network for classifying Fashion MNIST images using Keras. First, the Fashion MNIST data is loaded, reshaped into 784-dimensional vectors, and normalized. The labels are converted into one-hot encoded vectors to prepare for multi-class classification. The model consists of an input layer, a hidden dense layer with 128 neurons using ReLU activation, and an output layer with 10 neurons using softmax for classification. The model is compiled with the Adam optimizer, categorical cross-entropy loss, and accuracy as a metric. It is trained for 10 epochs with a batch size of 256, and validation is performed on the test set. Finally, the training and validation accuracy and loss are plotted over the epochs for visual comparison.