# Credit Card Image Classifier Using CNN Assignment

**Name: Bhaskar Upreti**

**Internship: AI/ML Internship**

**Date: 14/10/2025**

## Project objective:-

The objective of this assignment is to bulid the credit card image classifier that help us to identify the what credit card of it like visa, mastercard, Rupay, SBI etc. and it is made by the Convolutional Neural Networks (CNN).The model is trained by the the dataset of the image of different credit card and than test it and find the accuracy that how much it perfectly runs.I collected the images of 3 cards — Visa, Mastercard, and RuPay**.** I found the RuPay images by searching on Google Images, and the visa and Mastercard images I got from the SuperCardSet GitHub where the images were already available. I could only collect 15 images of RuPay, so I used 15 images for Visa and Mastercard also to keep the dataset balanced.I divided the data so that 12 images from each class were used to train the model and 3 images were used to test the model. I used TensorFlow to classify the images of the credit cards. I used the imageDataGenerator in TensorFlow and gave it the path of the train and test folders. It converted the image pixel values from 0–255 to 0–1 by dividing them by 255, because the pixel range is between 0 to 255.After that, I resized all images to 128×128**.** I used the batch size of 8 because the dataset is small, and batch size means how many images are processed at one time — here 8 images are processed together before updating the model weights. I used categorical class mode because we have 3 classes, and it is used when there are more than two categories.Then I imported the layers and models from TensorFlow and used the sequential model. I wrote the input size and used conv2d which checks the edges and patterns in the image. I used a (3,3) filter, which means it looks at a 3×3 area of pixels at a time. Then I used maxpolling2d, which finds the important parts of the image. I used it two times so that the model can learn all details of the images better.After that, I used flatten to convert the data from 2D to 1D so it can go into the dense layer. For compiling the model, I used the Adam optimizer**,** categorical_crossentropy as the loss function because there are 3 classes, and I checked accuracy as the metric. I trained the model for 15 epochs so that it runs faster but still learns the patterns.When I trained and tested the model, it gave an accuracy of 1.0 and a loss of 0.0708. This happened because the dataset is small, so the model overfitted — it memorized the training data instead of learning properly.Then I calculated the confusion matrix**,** precision**,** accuracy, and F1 score for the model:
Confusion Matrix:

```
[[2 0 1]
 [1 1 1]
 [0 2 1]]
```

Precision: 0.44
Accuracy: 0.44
F1 Score: 0.44

This shows that the model is not perfect because the dataset is small and overfitting occurred. I can improve it in the future by collecting more data or using data augmentation.