# INTERNSHIP REPORT

**Organization**: NullClass
**Internship Title**: Data Science Intern
**Intern Name**: Shreyash Shinde
**Duration**: 18-02-2025 to 18-04-2025
**Technologies Used**: Python, OpenCV, Keras, TensorFlow, Tkinter, PIL, datetime

## Introduction

This internship at NullClass focused on developing real-time machine learning applications using Python. The goal was to build and implement computer vision solutions ranging from image classification to age-gender prediction and gesture recognition. The experience emphasized end-to-end development – including dataset preparation, model training, GUI design, and real-time processing with logging and access control.

## TASK 1: Basic CNN Model for Image Classification

**Objective:**

To build a Convolutional Neural Network (CNN) model using Keras for classifying images into multiple categories.

**Process Logic:**

- Imported image processing and deep learning libraries such as OpenCV, TensorFlow, and Keras.
- Used ImageDataGenerator to preprocess and augment training images.
- Created a CNN architecture with Conv2D, MaxPooling2D, Flatten, and Dense layers.
- Compiled and trained the model using the Adam optimizer.

**Dataset:**

- Custom image dataset organized in train and test directories.
- Each folder labeled according to its class (e.g., "Cat", "Dog", etc.).
- Images resized and normalized for better training performance.

**Outcome:**

A trained CNN model capable of classifying custom images with decent accuracy.

**TASK 2: GUI-Based Image Classifier**

**Objective:**

To create a graphical interface for the CNN model developed in Task 1, allowing users to upload and classify images in real time.

**Process Logic:**

- Reused the trained CNN model.
- Developed a Tkinter GUI for user interaction.
- Integrated PIL and OpenCV for image handling and preprocessing.
- Enabled the GUI to load an image, preprocess it, and display the predicted label.

**Dataset:**

- Used the same dataset as Task 1 for model training.
- GUI tested with external images not seen during training.

**Outcome:**

Successfully created a user-friendly application that classifies uploaded images and displays the output label.

## TASK 3: Age and Gender Prediction with Logging

**Objective:**

To develop a real-time system that detects faces via webcam, predicts age and gender, and logs this information.

**Process Logic:**

- Loaded a pre-trained age and gender classification model.
- Used OpenCV to capture real-time webcam feed.
- Detected faces and predicted age and gender.
- Logged each prediction with timestamp into a CSV file.

**Dataset:**

- Based on pre-trained models trained on IMDb-WIKI or UTKFace datasets.

**Outcome:**

Created a live system capable of age and gender detection with real-time logging to maintain visitor data.

## TASK 4: Age & Gender Detection for Roller Coaster Entry

**Objective:**

To enhance the model from Task 3 by applying business logic for entry control based on age.

**Process Logic:**

- If predicted age < 13 or age > 60, person is "Not allowed".
- Restricted entries highlighted using **red boxes** on the webcam feed.
- All entries (allowed and not allowed) logged with timestamp, age, and gender.

**Dataset:**

- Same webcam-based input and pre-trained models as Task 3.

**Outcome:**

Built a smart entry monitoring system using computer vision to ensure safety and policy compliance at amusement rides.

## TASK 5: Sign Language Detection (Image & Video with Time Filter)

**Objective:**

To develop a CNN model to recognize sign language gestures and integrate it into a time-bound GUI system for image and video input.

**Process Logic:**

- Built a CNN to classify hand gestures representing sign language.
- Used ImageDataGenerator for data augmentation and training.
- Developed a Tkinter GUI with two options: Image Mode and Video Mode.
- Implemented time-based access control using the datetime module:
    - The application functions only between 6 PM and 10 PM.

**Dataset:**

- Custom sign language dataset with labeled folders for each alphabet (A, B, C, etc.).

**Outcome:**

A complete AI-powered sign language recognition app that restricts operation time and handles both image and video inputs.

## OVERALL LEARNINGS

- Built multiple CNN models for diverse use cases.
- Gained practical experience integrating ML models into GUI applications.
- Learned real-time video processing using OpenCV.
- Worked on logging, facial detection, and system control based on logic and time.
- Strengthened knowledge of libraries like TensorFlow, Keras, PIL, and Tkinter.

## CHALLENGES FACED

- Data preprocessing for real-time video feeds.
- Managing model performance with limited dataset size.
- Handling multiple UI elements and real-time feedback in GUI.
- Debugging time-restricted features and camera integration.

## CONCLUSION

This internship helped me gain hands-on experience in applying machine learning models to real-world applications. I learned how to preprocess data, train deep learning models, integrate them with graphical interfaces, and add intelligent features like logging and time restrictions. The structured tasks offered a well-rounded perspective on computer vision and AI deployment.