CHESS PROJECT:

In this project, I developed a machine learning model that predicts the **FEN (Forsyth-Edwards Notation)** score from an image of a chessboard. The FEN score is a way to represent the current state of a chess game using a single string. The goal was to automatically recognize the position of the pieces on a chessboard from an image and generate a valid FEN string.

The motivation behind this project was to create a tool that could convert any chessboard image, whether from an online match or a physical board, into a computer-readable format. This has applications in chess engines where players can analyse their games from that current state

I started by collecting a dataset of chessboard images, each labeled with its corresponding FEN notation. The first step in the pipeline was image preprocessing. Each chessboard image was resized to a standard size, say 224x224 pixels, so that all the images fed into the model were uniform. I also converted the images to grayscale, simplifying them into a single color channel, which reduces computational complexity and helps the model focus on the shapes and positions of the pieces instead of the color. Additionally, I normalized the pixel values to be between 0 and 1, which helps in stabilizing and speeding up the learning process. To handle the output, I used one-hot encoding for representing the chessboard

state.There are 64 squares on the chessboard, and each square can either be empty or contain one of 12 possible pieces (6 for white, 6 for black). This means that each square can be represented by a 13-dimensional one-hot encoded vector (12 for pieces, 1 for an empty square).
The entire board is represented by a flat one-hot encoded vector of size 832 (64 squares * 13 possible states).

This serves as the target label during model training.
So, the model's goal is to predict this 832-dimensional one-hot encoded vector, which represents the positions of the pieces.

Next, I designed the model using a Convolutional Neural Network (CNN).

The input to the model is the chessboard image (e.g., 224x224), which is passed through multiple convolutional layers. These layers are excellent at extracting spatial features from images—things like the positions of pieces, edges, and relative distances between the pieces.
After the convolutional layers, there's a flatten layer that converts the 2D spatial features into a 1D vector.
This 1D vector is then passed through several fully connected layers to learn more abstract patterns about the game state.
Finally, the output layer predicts the one-hot encoded vector that represents the entire chessboard state (832-dimensional).

I trained the model using supervised learning. The chessboard image was the input, and the one-hot encoded vector of piece positions was the target output.

Backpropagation was applied to adjust the model weights during training, and an optimizer like Adam was used to efficiently minimize the loss function.

Once trained, the model could take a new chessboard image as input, process it through the convolutional layers to detect piece positions, and finally output a one-hot encoded vector representing the entire board state. This vector could then be decoded back into the FEN string format.

In summary, this project involved predicting the FEN notation from chessboard images using deep learning. I preprocessed the images, used one-hot encoding to represent chessboard states, and trained a CNN to map images to these encodings. This project helped me deepen my understanding of CNNs, image classification, and how to encode structured data like chessboard states efficiently.

ADD ONS : 1. Informatica Journal Acceptance

CHALLENGES FACED:
Detecting small differences in piece positions was sometimes tricky, so I fine-tuned the architecture and added dropout layers to prevent overfitting