# Project Report: AI-Based Ludo Game Using Pygame and PyTorch

## Introduction

This project presents an AI-powered digital version of the popular board game Ludo. Developed using Python, the game leverages Pygame for its graphical interface and PyTorch for artificial intelligence. The goal is to allow players to compete against each other or AI agents in a simulated environment that mimics the traditional Ludo board game with engaging visuals and sound effects.

## Project Objectives

- Create an interactive Ludo board game interface.
- Implement all traditional game rules, including dice rolling, token movement, and token capturing.
- Integrate AI agents to enable single-player mode against computer-controlled opponents.
- Enhance user experience with sound and visual feedback.

## Technology Stack

- Programming Language: Python 3
- Graphics & UI: Pygame
- Artificial Intelligence: PyTorch
- Audio: Pygame Mixer
- Data Handling: Pickle
- Game Logic: NumPy, Random

## Directory Structure and Key Files

```
Ludo/
├── Ludo.py            # Main game script
├── requirements.txt   # Dependencies
├── info.pkl           # Saved model or game state
├── pics/              # Contains game images (board, dice, tokens)
├── sounds/            # Contains audio files (e.g., kill sounds)
└── .git/              # Git version control folder
```

- Ludo.py: Implements the game loop, user input handling, token logic, dice simulation, and AI integration.
- pics/: Includes visual assets for the board and tokens.
- sounds/: Contains .wav files for in-game sound effects.

- info.pkl: Likely used for storing serialized objects like AI model data or game statistics.
- requirements.txt: Lists required Python libraries.

## Features

- Full Ludo game simulation with four players (red, green, yellow, blue).
- Dice roll visualized with images from 1 to 6.
- Token kill and movement mechanics.
- Game state management and turn-based logic.
- Sound effects for immersive experience.
- AI integration using PyTorch for decision-making.

## Artificial Intelligence

The game includes a basic AI implementation using PyTorch, which allows agents to:
- Decide which token to move based on game state.
- Use simple rules or pre-trained models for decision-making.
- Simulate competition with human-like behavior.

The AI logic is embedded within the game script and interacts with the game environment during each agent's turn.

## Game Flow

- Initialization:
  - Load assets (images and sounds).
  - Initialize game screen and board.
- Game Loop:
  - Wait for player input or AI move.
  - Roll dice and update visuals.
  - Determine possible moves.
  - Execute token movement.
  - Check for token captures and home reach.
  - Change turn and repeat.
- Winning Condition:
  - A player wins when all four of their tokens reach the home area.

## Future Improvements

- Implement advanced AI with reinforcement learning.
- Add multiplayer mode over a network.
- Mobile or web version for broader accessibility.
- User profile management and score tracking.
- Theme customization and animated effects.

## Conclusion

This project successfully demonstrates the development of a classic board game enhanced with AI. It showcases the integration of game design, artificial intelligence, and multimedia elements using Python. The codebase serves as a foundational framework for further expansion into more complex game mechanics and smarter AI agents.

## Setup Instructions

1. Install dependencies:
   pip install -r requirements.txt

2. Run the game:
   python Ludo.py