**Project Title:** Autonomous Taxi Agent

Reinforcement Learning- ETE 3

**Objective:** Develop an intelligent agent using reinforcement learning to solve the Taxi-v3 environment from OpenAI Gym. The goal is for the taxi agent to pick up and drop off passengers at designated locations efficiently.

**Environment Description:**

* **Name:** Taxi-v3
* **Type:** Grid-based simulation with discrete states and actions.
* **Objective:**
  + Navigate a grid to pick up and drop off passengers.
  + Minimize the number of steps and maximize rewards.
* **State Space:** Finite set of states representing the taxi's location, passenger location, and destination.
* **Action Space:** Move North, South, East, West, Pick Up, Drop Off.

**Technologies and Libraries Used:**

* **Python**: Core programming language.
* **NumPy**: Numerical computations.
* **OpenAI Gym**: Provides the Taxi-v3 environment.
* **Matplotlib**: Visualization of performance metrics.
* **Pygame**: Supports enhanced rendering.

**Methodology:**

1. **Environment Initialization:**
   * Load the Taxi-v3 environment using gym.make('Taxi-v3').
   * Inspect the state and action spaces for setup.
2. **Q-Learning Implementation:**
   * Initialize a Q-table with dimensions equal to state and action spaces.
   * Define hyperparameters:
     + Learning rate (α): Controls the rate of Q-value updates.
     + Discount factor (γ): Determines the importance of future rewards.
     + Exploration rate (ε): Balances exploration and exploitation.
   * Iterate through episodes to:
     + Take actions based on the ε-greedy policy.
     + Update Q-values using the Bellman equation.
3. **Training and Evaluation:**
   * Train the agent over multiple episodes to optimize its policy.
   * Visualize training performance using metrics like:
     + Cumulative rewards.
     + Number of steps per episode.
   * Evaluate the trained agent by running it in the environment.
4. **Enhancements and Visualization:**
   * Install pygame, xvfb, and python-opengl for rendering.
   * Display environment transitions and actions.

**Key Results:**

* Trained the taxi agent to efficiently complete the pick-up and drop-off tasks.
* Optimized cumulative rewards and reduced episode lengths over time.

**Future Improvements:**

* Experiment with advanced algorithms like Deep Q-Learning.
* Explore additional environments to test the agent’s generalization capability.
* Enhance visualizations for better interpretability of the agent's actions.

**Conclusion:** This project demonstrates the application of Q-Learning to a discrete action and state space environment, achieving significant improvements in the agent's performance over training episodes.