# COVERPAGE

[**COVERPAGE 1**](#_ga4t2w7noa6n)

[**1. INTRODUCTION 1**](#_7wcz0zblognp)

[1.1. Background 1](#_396fdd3k537x)

[1.2. Problem Statement 1](#_ftjw50ndf7ab)

[**2. RESEARCH QUESTION 1**](#_amymvexzmwj5)

[**3. AIM AND OBJECTIVES 2**](#_l1pkkm8l9rtj)

[3.1. Aim 2](#_tjznc7c7l5nw)

[3.2. Objectives 2](#_dfpymxqlhi7n)

[**4. INTENDED USER GROUP & REQUIREMENTS 2**](#_o4ud6sfe2ye1)

[**5. MINI-LITERATURE REVIEW 3**](#_73ax4om47zxe)

[**6. RESEARCH METHODOLOGY 3**](#_vlj5zz41at1e)

[6.1. Approach 3](#_jew8il4x4zfi)

[6.2. Simulator and Environment Design 4](#_yf8uavyxcg4r)

[6.3. Dataset Description 4](#_qmbft7oc8n3x)

[6.3. Statistical Testing 5](#_h4jv0atanhvg)

[**7. PROJECT PLAN/ TIMELINE 5**](#_wo1ypj7280op)

[**8. RISK REGISTER 5**](#_it1j8dex3bmy)

[**9. SYSTEM REQUIREMENTS & DELIVERABLES 6**](#_up52ykvt4v84)

[9.1 System Requirements 6](#_ug2e0y5rsicj)

[9.2. Final Deliverables 6](#_4las2xa1567g)

[**REFERENCES 7**](#_verenej5rqpr)

# 

# INTRODUCTION

## 1.1. Background

The origin of Roulette dates back to 18-th century. Since then, it has attracted a lot of attention from mathematicians and gamblers. The wheel is theoretically a game of pure chance (Kavanagh, 2009). The house has a built-in advantage. It is 2.70% on a European wheel and 5.26% on an American one (Jensen, 1998). This makes it hard to beat with fixed betting strategies. But in recent times there has been significant advancement in reinforcement learning (RL) algorithms which learn optimal behavior by interacting with an environment (Padakandla, 2021). This raises the question of whether an adaptive agent can extract practical value in the short run before regression to the mean erodes its gain.

This project investigates whether reinforcement learning particularly, Proximal Policy Optimisation (PPO) can be used to build a dynamic roulette betting strategy that can get better performance in constrained, short-term settings compared to traditional methods (Schulman et al., 2017).

## 1.2. Problem Statement

Classic betting systems (Martingale, D’Alembert, Fibonacci) adjust stakes mechanically and ignore the past state of the wheel (Ethier & Hoppe, 2019). As a result, these approaches do not take advantage of temporary biases, wheel flaws, or table-specific payment regulations. In contrast, casino surveillance and regulation make physical wheel prediction infeasible (Boss & Zajic, 2010). The challenge is to create, train, and evaluate PPO algorithm based agents that (i) conform to European roulette rules and return payouts, (ii) change stakes and bet types in real time and (iii) maximize cumulative return within reasonable bankroll and table-limit restrictions.

# RESEARCH QUESTION

*“Can a Proximal Policy Optimisation (PPO) agent trained in a constrained, realistic roulette simulation environment outperform traditional static betting systems in terms of short-term cumulative return and risk-adjusted performance?”*

# AIM AND OBJECTIVES

## 3.1. Aim

The aim of this study is to develop and evaluate Proximal Policy Optimisation-based betting strategy for roulette which adapts in real-time and improves short-term performance under realistic constraints.

## 3.2. Objectives

The objectives of the project are presented below:

1. To create the design of a realistic European roulette simulator and implement it.
2. To integrate table rules, bet types, payout structures, bankroll limits, and spin horizons
3. To implement a static betting system like Margingale for benchmarking.
4. To create and train a PPO agent using environment feedback and engineered features.
5. To perform proper evaluation of the PPO agent across multiple metrics including risk of ruin, drawdown and cumulative return.
6. To perform comparison of the results obtained from PPO agents with baseline strategies via rigorous statistical testing.
7. To finally package the simulator and PPO framework into a reproducible research toolkit and document all processes and results in the final dissertation report.

# 4. INTENDED USER GROUP & REQUIREMENTS

*Table I: Intended User Groups, Their Requirements, and Anticipated Benefits*

| User Group | Requirements | Benefits |
| --- | --- | --- |
| AI researchers & developers | Access to PPO training scenarios and replicable benchmarks. | Useful testbed for RL algorithm evaluation |
| Casino analysts | Insight into betting behavior and system performance. | Tools for risk management and bias identification |
| Educators / students | Simplified PPO-based implementation with visual results | Resources for RL and game theory teaching |
| General public | Interactive demo using virtual cash. | Exploration of advanced strategies without real risk |

# 5. MINI-LITERATURE REVIEW

Conventional roulette betting systems, such as Martingale and D'Alembert, are based on pre-determined stake progressions and are unrelated to the history of previous results. Rigorous statistical investigation confirms that these progression-based systems optimize variance and maintain the adverse expected value inherent to fair roulette wheels. For example, a conclusive simulation analysis of doubling schemes demonstrates that while Martingale is profitable in the short run, the long-term expected return is negative and catastrophic losses become inevitable when violating table limits or running out of bankrolls (Pflaumer, 2019).

Proximal Policy Optimisation (PPO) algorithm is proposed in paper Schulman et al. (2017). PPO is a modern RL algorithm which improves training stability and efficiency by limiting policy updates. According to the paper, the algorithm balances exploration and exploitation using clipped goal functions, making it appropriate for noisy and unpredictable situations (Schulman et al., 2017).

Reinforcement learning has been successfully used in complex games like GO and backgammon (Silver et al., 2017; Tesauro, 1004). Silver et al.,(2017), was able to achieve superhuman performance in Go through tabula rasa reinforcement learning. Another paper Tesauro, 1994 applied temporal-different learning to backgammon. Despite these successful implementations, its application in games, there is limited research that has explored the implementation of RL in case of gambling settings like Roulette. Although less directly related, Zhao (2020) showed that RL can be effectively used for dynamic blackjack betting within regulatory constraints. This highlighted the potential of RL algorithms in stochastic environments. However, there are not many studies that have applied PPO to a full roulette simulation environment with comparative analysis against traditional strategies.

Thus, this project builds on the foundational work of PPO in gaming environments, extending it to roulette with realistic parameters and benchmarking.

# 6. RESEARCH METHODOLOGY

## 6.1. Approach

The research will use a simulation-based methodology. This method will safely and ethically evaluate performance of strategy in a controlled setting (Cheng et al., 2014). In order to provide a realistic and comprehensive testbed for PPO training, the simulation will replicate real-world roulette rules and limits.

## 6.2. Simulator and Environment Design

**Game Environment**

The simulator will be created in Python. It will simulate European roulette (single-zero) and supports various types of bets like dozen, red/black, odd/even, street corner, split, column, and straight. It monitors spin histories, bet results, and bankroll swings in addition to enforcing table betting restrictions.

**Baseline Strategies**

* Martingale
* Fibonacci
* Flat Betting

**PPO Agent Architecture**

* Built using deep learning libraries like Keras and PyTorch and trained using Stable PPO implementation.
* State Space: Recent outcome history, bet history, current bankroll and derived features (e.g., hot/cold numbers).
* Action Space: Bet type, target numbers or colors, and bet amount (modified to comply with casino regulations) are chosen.
* Reward Function: Penalties for excessive risk or bet concentration are applied on the net change in bankroll.

## 6.3. Dataset Description

In this project, synthetic data will be generated via simulation.This will involve millions of spin results, wagers, and reward trajectories. For training, approximately one million spins will be produced as the agent learns to optimise its betting strategy. The spin outcome, the bet selected, the stake amount, the instant payout received, and the updated bankroll status all will be included in each training record.

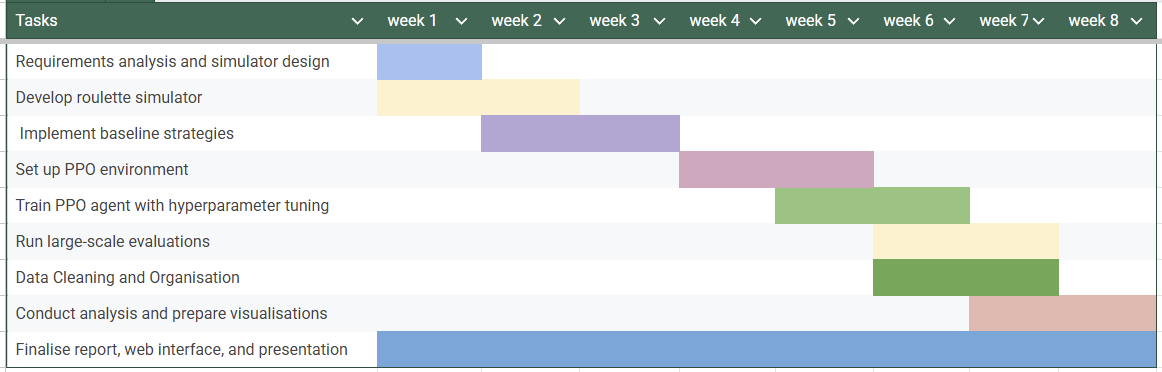
For evaluation, about 1,000 independent simulations of 5,000 spins each will run. It will cover both PPO and baseline strategies for comparison.

Statistics such as total return, drawdown, volatility, risk of ruin, and behavior of the agent will be monitored. Data is saved with information for repeatability in CSV and binary formats which will work with the libraries like Pandas, PyTorch and Keras. This internally generated data will ensure transparency, control and reliable statistical analysis under different environments.

## 6.3. Statistical Testing

Paired t-tests and Wilcoxon signed-rank tests will be used to validate improvements over baseline strategies (Katakam, 2025).

# 7. PROJECT PLAN/ TIMELINE



# 8. RISK REGISTER

| **Risk** | **Likelihood** | **Impact** | **Mitigation** |
| --- | --- | --- | --- |
| Convergence failure for the PPO agent | Medium | High | Make use of curricular learning and reward structuring (Pendyala et al., 2024). |
| A bug in the payout algorithm of the simulator | Medium | Medium | Comparing unit testing to theoretical probabilities |
| Not enough processing power | Low | Medium | Use university GPU resources or cloud alternatives |
| lack of statistically significant results | Medium | Medium | Increase number of simulation runs; add robustness checks |

# 9. SYSTEM REQUIREMENTS & DELIVERABLES

## 9.1 System Requirements

Hardware

* CPU: Quad-core or higher
* GPU: Optional but recommended (e.g., NVIDIA RTX 3060 or equivalent)
* RAM: 16 GB or more

Software

* Python 3.12
* PyTorch or Keras
* Stable Baselines3 (PPO implementation)
* Jupyter, Pycharm, (for demo)

## 9.2. Final Deliverables

1. Completely documented European Roulette simulator with realistic constraints.
2. Baseline betting strategies implemented with benchmarking scripts.
3. A trained PPO agent with saved models and training history.
4. Notebooks for evaluating statistical data and making visual comparisons.
5. A well-documented and written dissertation report.

# REFERENCES

Boss, D. J., & Zajic, A. W. (2010). *Casino security and gaming surveillance*. CRC Press.

Cheng, A., Auerbach, M., Hunt, E. A., Chang, T. P., Pusic, M., Nadkarni, V., & Kessler, D. (2014). Designing and conducting simulation-based research. *Pediatrics*, *133*(6), 1091-1101.

Ethier, S. N., & Hoppe, F. M. (2019). Teaching a university course on the mathematics of gambling. *arXiv preprint arXiv:1911.03008*.

Jensen, M., 1998. *Secrets of Winning Roulette*. Cardoza Publishing.

Katakam, A. (2025). Continuous variable analyses: T-test, Mann–Whitney test, and Wilcoxon signed-rank test. In *Translational Gastroenterology* (pp. 135-136). Academic Press.

Kavanagh, T. M. (2009). Roulette and the Ancien Régime of Gambling. *Nottingham French Studies*, *48*(1), 1-13.

Padakandla, S. (2021). A survey of reinforcement learning algorithms for dynamically varying environments. *ACM Computing Surveys (CSUR)*, *54*(6), 1-25.

Pendyala, A., Atamna, A., & Glasmachers, T. (2024, August). Solving a Real-World Optimization Problem Using Proximal Policy Optimization with Curriculum Learning and Reward Engineering. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 150-165). Cham: Springer Nature Switzerland.

Pflaumer, P. (2019). A statistical analysis of the roulette martingale system: examples, formulas and simulations with R.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O., 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, *38*(3), 58-68.

Zhao, M. (2020). Continuous Blackjack: Equilibrium, Deviation and Adaptive Strategy. *arXiv preprint arXiv:2011.10315*.