

A Project Report On

Option Selling Tracker

Submitted in partial fulfillment of the requirement for the
award of the degree

Bachelor of Computer Application
BCA

Academic Year 2025 - 26

Bhavya Mehta
92300527128

Jeel Patel
92300527015

Aagnik Goswami
92300527012

Internal Guide
Prof. Dipak Thanki



Marwadi
University
Marwadi Chandarana Group





Marwadi
University
Marwadi Chandarana Group



Faculty of Computer Applications (FCA)

Certificate

This is to certify that the project work entitled
Option Selling Tracker
submitted in partial fulfillment of the requirement for
the award of the degree of
Bachelor of Computer Application
BCA
of the
Marwadi University
is a result of the bonafide work carried out by
Bhavya Mehta (92300527128)
Jeel Patel (92300527015)
Aagnik Goswami (92300527012)

during the academic year 2025-26

Prof. Dipak Thanki

Dr. Sunil Bajaja

Prof. (Dr)
Sridaran Rajagopal

Faculty Guide

HOD

Dean

DECLARATION

We hereby declare that this project work entitled **Option Selling Tracker** is a record done by me.

We also declare that the matter embodied in this project is genuine work done by me and has not been submitted whether to this University or to any other University / Institute for the fulfillment of the requirement of any course of study.

Place : Rajkot

Date : 18/07/2025

Bhavya Mehta (92300527128) Signature : BHAVYA

Jeel Patel (92300527015) Signature : JEEL

Aagnik Goswami (92300527012) Signature : AAGNIK

CONTENTS

Chapters	Particulars	Page No.
1	SYNOPSIS	1
2	PREAMBLE General Introduction Module description	2-3
3	TECHNICAL DESCRIPTION Hardware Requirement Software Requirement	4
4	SYSTEM DESIGN AND DEVELOPMENT <ul style="list-style-type: none">• Flowchart• Use Case Diagram• Data Flow Diagram – Level 0• Data Flow Diagram – Level 0 Class Diagram Screen Design & Coding	5-15
5	CONCLUSION	16
6	LEARNING DURING MINI PROJECT	17-18
7	BIBLIOGRAPHY Online References Offline References	19

SYNOPSIS

Project Title: Option Selling Tracker – A CSV-Based Trade Management Tool Using Python and Tkinter

This project is a desktop GUI application built with Python and Tkinter, designed to simplify and streamline tracking of option selling trades. Using a CSV file as the core data store, the system enables users to add, update, search, delete, and clear trade records efficiently. It supports detailed trade inputs including hedging information and maintains uniqueness of trades using composite keys. The tool is particularly useful for retail traders who require an easy-to-use, standalone trade management application without relying on spreadsheets or complex databases.

PREAMBLE

General Introduction:

The stock market is increasingly accessible to retail traders, many of whom engage in option selling strategies as a means to generate regular income. Option selling involves writing call or put options to collect premiums, a process requiring disciplined record-keeping, trade management, and risk control. Currently, many traders rely on spreadsheets or manual logs for trade tracking, which can be inefficient and error-prone.

To overcome these issues, this project introduces a Python-based Option Selling Tracker—a desktop GUI application developed using Tkinter. It provides users with an intuitive interface to record, search, update, delete, and manage trades stored in CSV files. Key features include support for hedging positions, enforcement of trade uniqueness through composite keys, and streamlined navigation for essential trade management operations.

Module Description:

The system is organized into the following functional modules:

- 1. Add Trade:**

Enables the user to input a new option selling trade, including the option to add associated hedge positions. Trade uniqueness is ensured using a composite key combining Strategy Name, Trade Date, and Instrument.

- 2. Display Trades:**

Presents all stored trades in a dynamic, scrollable table by reading data from the CSV file.

- 3. Search Trade:**

Facilitates searching for specific trades based on the composite key.

4. Update Trade:

Allows editing any trade detail with a user-friendly interface where fields can be optionally skipped.

5. Delete Trade:

Provides functionality to remove a selected trade identified uniquely by keys.

6. Clear All Records:

Implements a confirmation-based mechanism to delete all trade records in bulk.

Each module aims to offer an efficient, reliable, and user-friendly experience tailored to the needs of retail option sellers who prefer fast, straightforward trade tracking without relying on complex databases or online services.

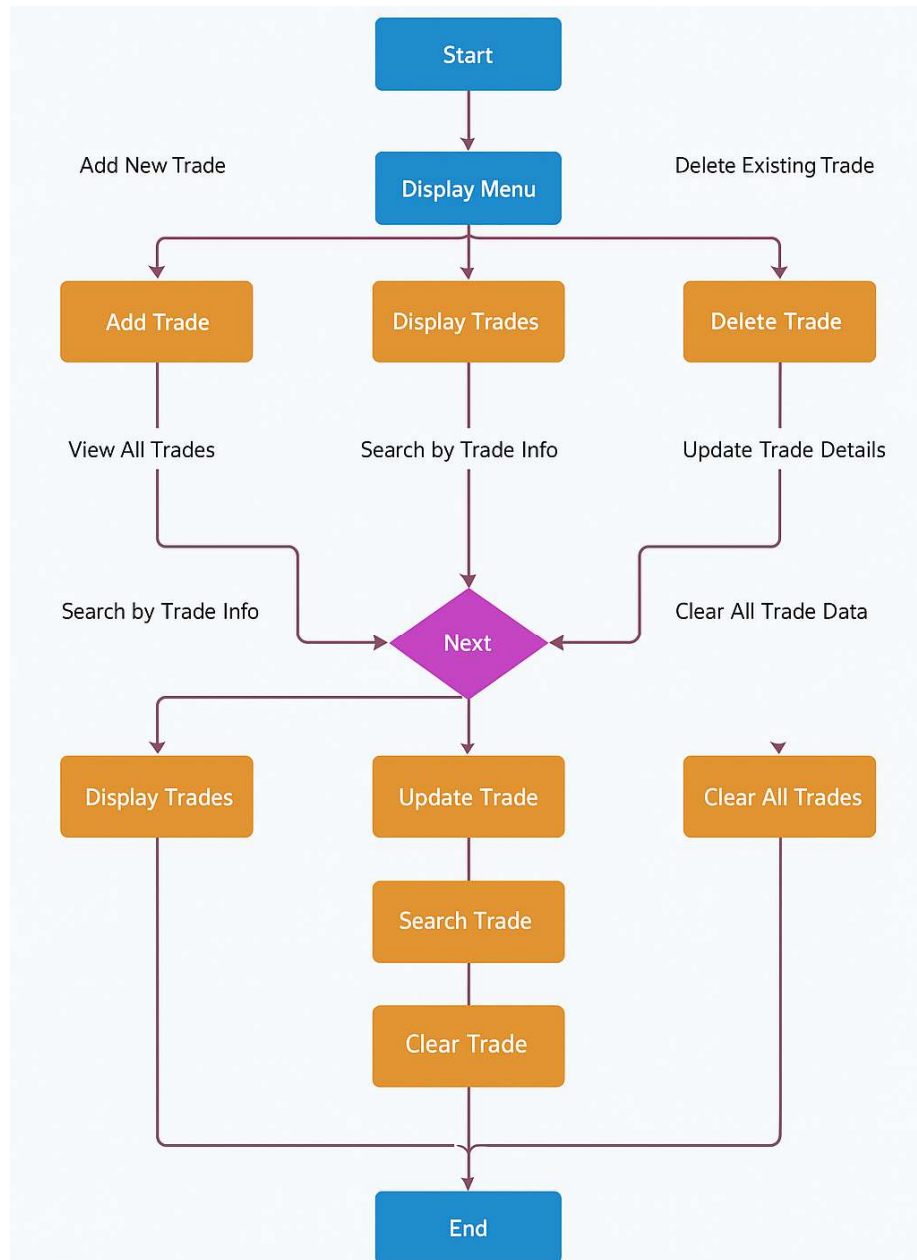
TECHNICAL DESCRIPTION

- **Hardware Requirement:**
 - Any system capable of running Python (Windows, macOS, Linux)
 - At least 512MB RAM (minimal requirement)
 - Minimal disk storage for CSV files

- **Software Requirement:**
 - Python 3.x (recommended latest stable version) with Tkinter module (usually included by default)
 - Any text editor or IDE for code modifications
 - Graphical desktop environment to run the Tkinter GUI application
 - No command-line interaction is required, but Python should be executed in a shell or terminal to launch the app

SYSTEM DESIGN AND DEVELOPMENT

1. Flowchart:



[Flow chart of Option Selling Tracker]

2. Use Case Diagram:

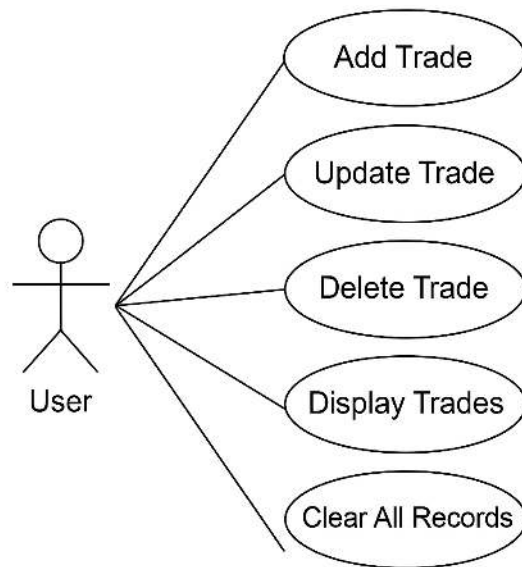


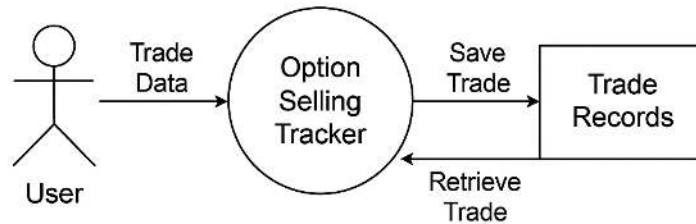
Figure: Use Case Diagram of Option Selling Tracker

The use case diagram illustrates the interaction between the user and the system functionalities of the Option Selling Tracker. The single actor, *User*, interacts with the system through six primary use cases:

- **Add Trade:** Allows entering a new option selling trade along with optional hedge details.
- **Update Trade:** Enables editing existing trade information with the ability to skip fields.
- **Delete Trade:** Removes a specific (selected) trade.
- **Display Trades:** Shows all the trades stored in the CSV file.
- **Search Trade:** Finds a specific trade using a unique key.
- **Clear All Records:** Deletes all stored trade records after confirmation.

This diagram helps visualize the user-driven functionalities in the system.

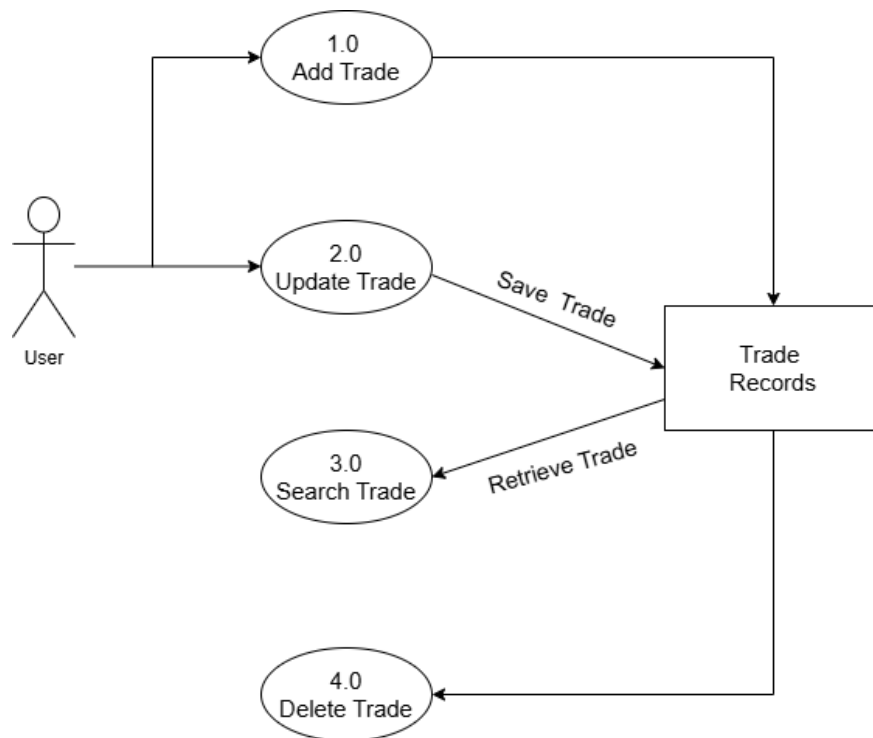
3. Data Flow Diagram – Level 0:



Data Flow Diagram (Level 0)

The Level 0 DFD gives a high-level overview of the Option Selling Tracker system. It depicts the interaction between the **User** and the **system**, where all user requests (add, update, delete, search) are processed by a central module. The **system** accesses and updates the **Trade Records** stored in a CSV file and returns the result to the user. This abstraction helps in understanding the overall system workflow without going into implementation-level details.

4. Data Flow Diagram – Level 1:



Data Flow Diagram (Level 1)

The Level 1 DFD breaks down the main system into four specific processes:

- **1.0 Add Trade** – Accepts new trade entries and stores them.
- **2.0 Search Trade** – Retrieves a trade based on strategy name, trade date, and instrument.
- **3.0 Update Trade** – Modifies specific fields of a selected trade.
- **4.0 Delete Trade** – Deletes a trade record from storage.

All modules interact with the **Trade Records** data store (CSV file). The **User** acts as the external entity initiating all processes. This diagram helps in understanding the internal modules and their data flow more precisely.

5. Class Diagram:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Strategy Name	Trade Date	Instrument	Strike Price	Buy/Sell	Expiry Date	Type	Lots	Entry Price	Exit Price	Hedged Strike Price	Hedged Buy/Sell	Hedged Entry Price	Hedged Exit Price	Margin Used	Holding Period	P&L
2																	
3																	
4																	
5																	

6. Screen Design & Coding

Screen Design (CLI)

The project features a graphical user interface built with Python's Tkinter library, providing an intuitive and user-friendly experience.

Upon launching the application, the user is presented with a **tabbed interface** containing different functional panels corresponding to all major features of the Option Selling Tracker:

Adding a Trade

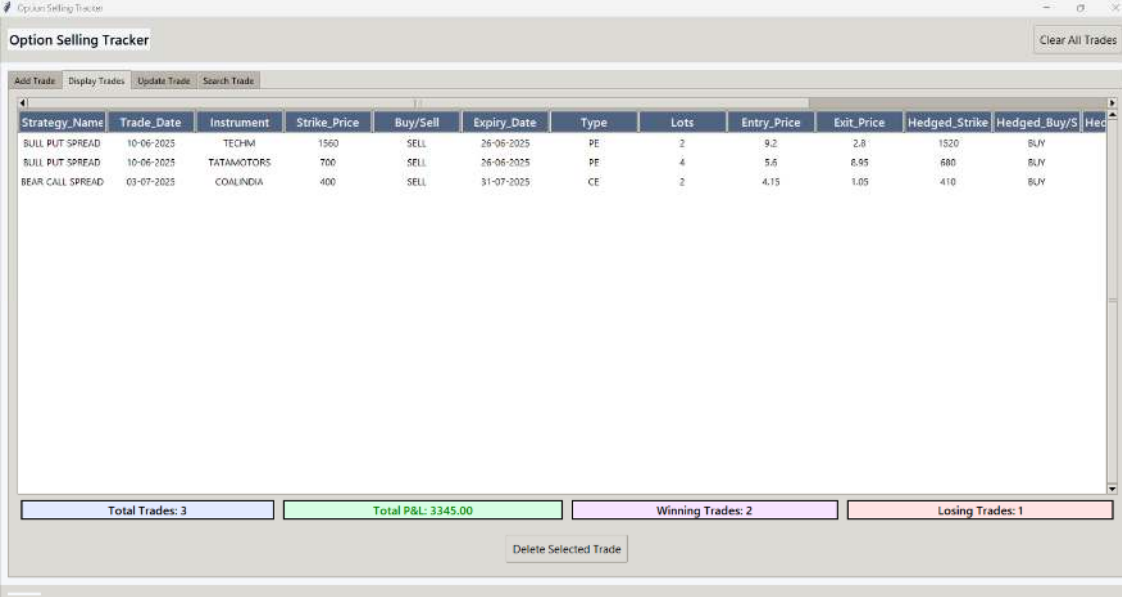


The screenshot shows the 'Option Selling Tracker' application window. The 'Add Trade' tab is selected. The form contains the following fields:

- Strategy Name:
- Trade Date (YYYY-MM-DD):
- Instrument:
- Strike Price:
- Buy/Sell:
- Expiry Date (YYYY-MM-DD):
- Type (CE/PE):
- Lots:
- Entry Price:
- Exit Price:
- Have you hedged your position? (y/n):
- Hedged Strike Price:
- Hedged Buy/Sell:
- Hedged Entry Price:
- Hedged Exit Price:
- Margin Used:
- Holding Period (Days):
- P&L:

At the bottom of the form is an 'Add Trade' button. In the top right corner of the window is a 'Clear All Trades' button.

Display Trades



The screenshot shows the 'Option Selling Tracker' application window with the 'Display Trades' tab selected. It displays a table of trades with the following data:

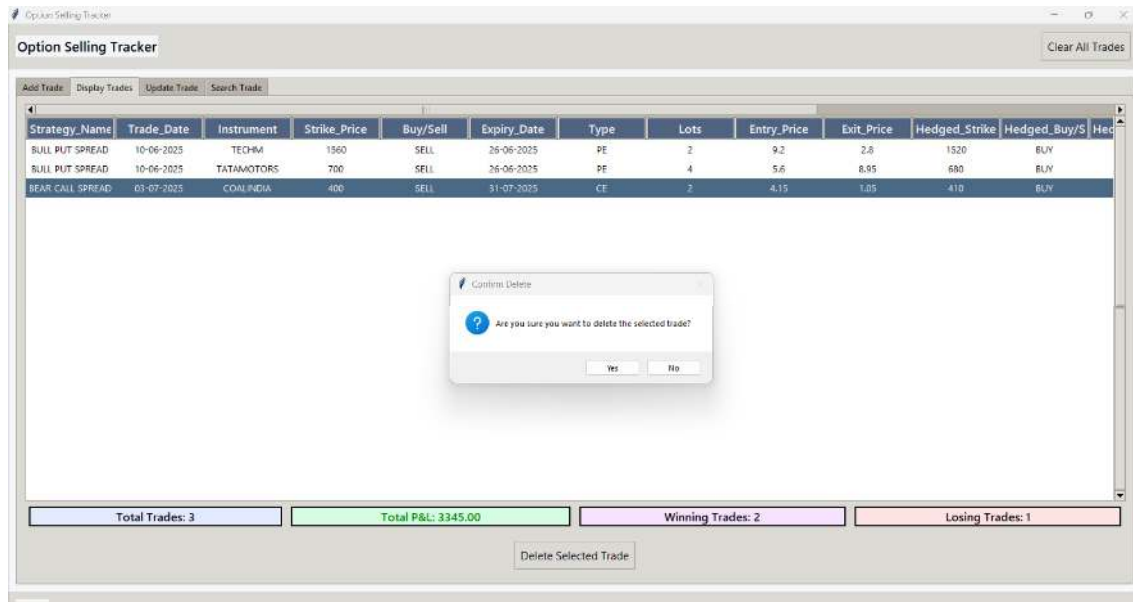
Strategy Name	Trade Date	Instrument	Strike Price	Buy/Sell	Expiry Date	Type	Lots	Entry Price	Exit Price	Hedged Strike	Hedged Buy/S	Hedged
BULL PUT SPREAD	10-06-2025	TECHM	1560	SELL	26-06-2025	PE	2	9.2	2.8	1520	BUY	
BULL PUT SPREAD	10-06-2025	TATAMOTORS	700	SELL	26-06-2025	PE	4	5.6	8.95	680	BUY	
BEAR CALL SPREAD	03-07-2025	COALINDIA	400	SELL	31-07-2025	CE	2	4.15	1.05	410	BUY	

Below the table, there is a summary bar with the following information:

- Total Trades: 3
- Total P&L: 3345.00
- Winning Trades: 2
- Losing Trades: 1

A 'Delete Selected Trade' button is located at the bottom center of the window.

Delete Trade



Update Trade



Option Selling Tracker

Clear All Trades

Add Trade Display Trades Update Trade Search Trade

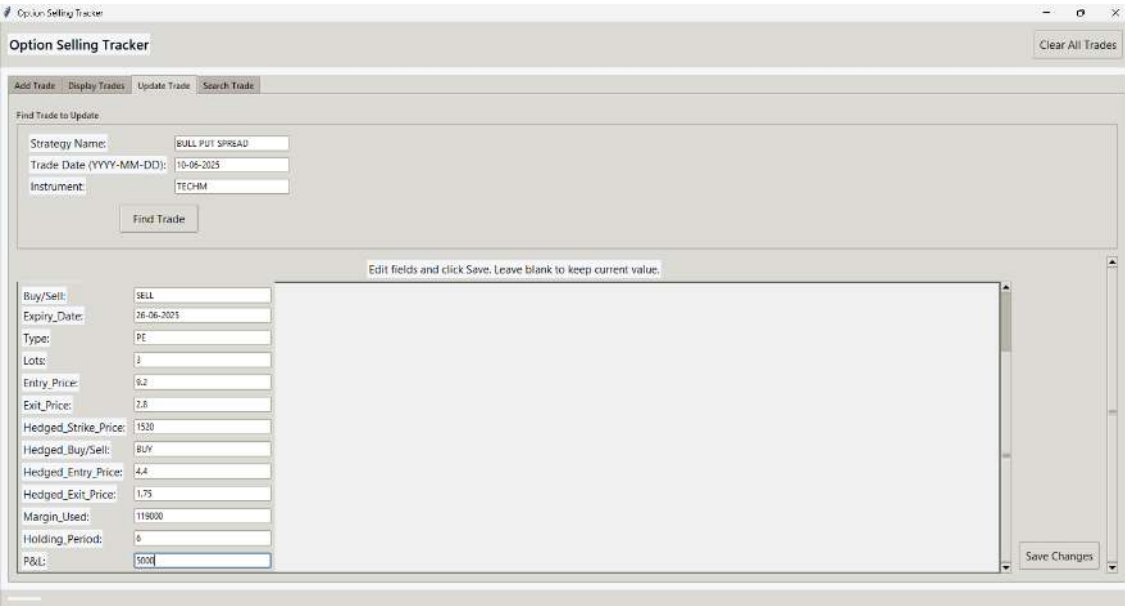
Find Trade to Update

Strategy Name:

Trade Date (YYYY-MM-DD):

Instrument:

Find Trade



Option Selling Tracker

Clear All Trades

Add Trade Display Trades Update Trade Search Trade

Find Trade to Update

Strategy Name:

Trade Date (YYYY-MM-DD):

Instrument:

Find Trade

Edit fields and click Save. Leave blank to keep current value.

Buy/Sell:	<input type="text" value="SELL"/>
Expiry Date:	<input type="text" value="26-06-2025"/>
Type:	<input type="text" value="PE"/>
Lots:	<input type="text" value="3"/>
Entry Price:	<input type="text" value="9.2"/>
Exit Price:	<input type="text" value="7.8"/>
Hedged Strike Price:	<input type="text" value="1530"/>
Hedged Buy/Sell:	<input type="text" value="BUY"/>
Hedged Entry Price:	<input type="text" value="4.4"/>
Hedged Exit Price:	<input type="text" value="1.75"/>
Margin Used:	<input type="text" value="119000"/>
Holding Period:	<input type="text" value="6"/>
P&L:	<input type="text" value="5000"/>

Save Changes

Search Trade



The screenshot shows the 'Option Selling Tracker' application window. At the top, there are tabs for 'Add Trade', 'Display Trades', 'Update Trade', and 'Search Trade'. The 'Search Trade' tab is active. Below the tabs, there is a 'Search Trade By' section with three input fields: 'Strategy Name' (containing 'BULL PUT SPREAD'), 'Trade Date (YYYY-MM-DD)' (containing '10-06-2025'), and 'Instrument' (containing 'TATAMOTORS'). A 'Search' button is located below these fields. Below the search fields, the results of the search are displayed in a text area, showing details for the 'BULL PUT SPREAD' strategy, including trade date, instrument, strike price, buy/sell type, expiry date, type, lots, entry price, exit price, hedged strike price, hedged buy/sell, hedged entry price, hedged exit price, margin used, holding period, and P&L.

Option Selling Tracker

Clear All Trades

Add Trade Display Trades Update Trade Search Trade

Search Trade By

Strategy Name: BULL PUT SPREAD

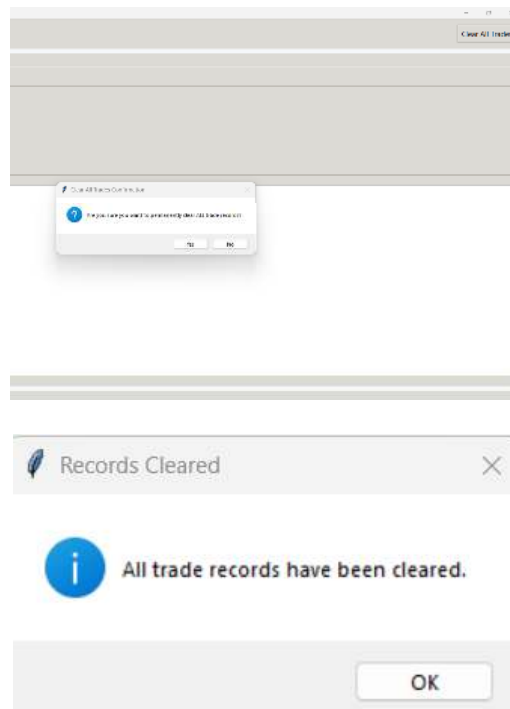
Trade Date (YYYY-MM-DD): 10-06-2025

Instrument: TATAMOTORS

Search

Strategy Name: BULL PUT SPREAD
Trade Date: 10-06-2025
Instrument: TATAMOTORS
Strike Price: 700
Buy/Sell: SELL
Expiry Date: 26-06-2025
Type: PE
Lots: 4
Entry Price: 5.6
Exit Price: 8.95
Hedged Strike Price: 680
Hedged Buy/Sell: BUY
Hedged Entry Price: 2.75
Hedged Exit Price: 4.3
Margin Used: 102000
Holding Period: 2
P&L: -3960

Clear All Trade



Coding

The project is implemented in Python utilizing core modules and features as follows::

- **Modules Used:** csv, os
These modules facilitate file handling and persistent data storage.
- **File Handling:**
All trade records are stored in a CSV file named option_selling_tracker.csv using Python's csv module for easy reading and writing of structured data.
- **Functions Implemented:**
 - add_trade() – Adds a new option selling trade, including optional hedge details.
 - display_trades() – Loads and displays all trades stored in the CSV file in a tabular format.
 - delete_trade() – Deletes an existing trade uniquely identified by Strategy Name, Trade Date, and Instrument.
 - update_trade() – Updates individual fields of a selected trade with the ability to keep existing values.
 - search_trade() – Searches for a particular trade record based on unique composite keys.
 - clear_all_records() – Permanently clears all stored trade records after user confirmation.
 - initialize_file() – Checks for existence of the CSV file and initializes it with appropriate headers if missing.
- **Program Structure:**
The application follows a modular design pattern, with distinct functions and UI sections dedicated to specific operations. This approach improves maintainability, readability, and extensibility of the codebase.

CONCLUSION

- The Option Selling Tracker is a lightweight yet effective **Python-based GUI tool** designed to help traders efficiently manage their option selling records without reliance on complex database systems. It leverages Python's csv module to store and manipulate structured trading data in CSV format, ensuring data portability and simplicity.
- Features like hedging support, preventing duplicate trades, and flexible update and search functionalities enhance both accuracy and usability of the system.
- This project deepened our understanding of Python's file handling, data validation, conditional logic, and GUI design with Tkinter. It exemplifies how a modular, well-structured application can provide significant practical utility while remaining simple.
- Future improvements could involve implementing advanced analytics for insights on P&L, integrating live market APIs for real-time data, or migrating from CSV to relational databases for enhanced scalability and performance.

LEARNING DURING MINI PROJECT

During the course of the Mini Project, We acquired significant technical and practical skills in programming and financial markets. The key areas of learning are summarized below:

Technical Learnings:

- **Python Programming:**
 - Gained hands-on experience writing modular, structured, and readable Python code implementing core concepts such as conditional statements, loops, user input handling, and function-based program structure.
- **CSV File Handling:**
 - Understood reading from, writing to, and updating CSV files using Python's csv module while maintaining data integrity and uniqueness of trade records.
- **Data Validation:**
 - Developed logic to prevent duplicate trades by verifying composite keys (Strategy Name + Trade Date + Instrument), and built a flexible update mechanism enabling partial record editing.
- **CLI-Based User Interface:**
 - Designed and implemented a user-friendly graphical interface based on Tkinter, facilitating straightforward navigation and improving user experience.

Domain-Specific Learnings (Option Selling & Trading):

- **Trade Record-Keeping:**
 - Recognized the criticality of maintaining accurate trade records for performance review and compliance, and learned key trade parameters like strike price, entry/exit price, margin used, P&L, and holding period.
- **Hedging Strategies:**
 - Gained insights into hedged positions in option selling as a risk mitigation technique and incorporated this understanding into project features.
- **Trade Analysis:**
 - Improved ability to analyze trades through search, display, and reporting functionalities aimed at detailed performance assessment.

Project Development Skills:

- **System Design:**
 - Applied structured design techniques, including data flow diagrams (DFD) and flowcharts, to conceptualize and visualize system workflow before coding.
- **Testing & Debugging:**
 - Executed thorough iterative testing and debugging processes, ensuring reliability and correctness of all functional features.
- **Documentation:**
 - Learned to document the entire project lifecycle professionally, including preparation of synopses, design diagrams, and user manuals.

BIBLIOGRAPHY

Online References

1. [W3Schools – Python File Handling](#)
Referred for file operations and input/output techniques.
2. [GeeksforGeeks – Python Projects](#)
Used for getting inspiration and coding patterns for CLI-based Python applications.
3. [NSE India – Option Trading Strategies](#)
Referred to understand real-world trading strategies and hedging examples.
4. [Investopedia – Options Selling Explained](#)
Referenced to understand the fundamental concepts and mechanisms of options selling.
5. [Real Python – CSV File Operations](#)
Used for best practices and examples on CSV file handling in Python.

Offline References

1. **Lecture Notes from Option Selling Training Class**
Personal handwritten notes taken during the stock market classes.
2. College lecture handouts for Python programming and CLI development.
3. Mini project journal provided by faculty, offering guidance on system design diagrams and documentation practices.