// CURRICULUM · INDEX

Six modules. One desk.

Coding · Platforms · Backtesting · Data · ML · Ops.
181 pages of honest material, free to read.

06	181	40+	00
MODULES	PAGES	FRAMEWORKS	FEES

NEXURAL · SAGE IDEAS LLC
PUBLISHED 2026 · PERPLEXITY COMPUTER

// overview

What's in the curriculum.

Six PDF modules and six matching web pages. They cover the path from writing your first Python script to running a desk with a kill-switch and a postmortem template. Every page assumes you are doing this with your own money, on your own time, with no firm behind you.

The material is opinionated. Where the industry disagrees we pick a side, say why, and link the dissenting sources so you can argue with us.

MODULE	PAGES	ONE-LINER
01 · Coding Fundamentals	31	The Python, git, and shell habits the rest of the curriculum assumes.
02 · Platform Coverage	14	An honest survey of the six platforms most people actually use.
03 · Backtesting	36	How to design a backtest that does not lie to you.
04 · Data Engineering	28	Bars, ticks, vendors, alignment, and the boring layer everything else rides on.
05 · Machine Learning	40	Models that survive contact with markets, and how to tell which do not.
06 · Operations, CLI & Recovery	32	The desk is the system. Ten procedures, six incidents, six trees, twelve monitors, twenty gates.

// reading order

How to read this.

The default path

Read in order: 01 → 02 → 04 → 03 → 05 → 06. Coding first, then a platform survey, then data engineering, then backtesting, then machine learning, then operations. The order matters: backtesting is hard to do honestly without a clean data layer, and ML is harder still without an honest backtest.

If you already write Python

Skim 01 and jump to 02 + 04 in parallel. They are the platform and data layers respectively, and reading them together makes the trade-offs clearer.

If you only care about running a strategy

Read 06 (Operations) first to see what running a strategy actually entails. Then come back to 01-05 to fill in the parts you do not yet have.

Pace

About six hours per module is a comfortable read with the worked examples. Skimming, three. Doing the exercises end to end, roughly two evenings each. The full curriculum is a long weekend or four weekday evenings if you skip the exercises.

// module · 01

Coding Fundamentals.

The Python, git, and shell habits the rest of the curriculum assumes.

PAGES	PDF	WEB
31	/pdfs/coding-fundamentals.pdf	/learn/automation/coding-fundamentals

What you'll learn

· Pure functions and how to keep side effects on the edges

· Type hints, dataclasses, and reading other people's code

· Project structure: src layout, tests, and one main.py

· git workflow: branches, PRs, and the rare force-push exception

· Shell loops you actually need — find, xargs, jq, awk

Prerequisites

None — start here if you have not written Python in the last month.

Leads into

Data Engineering and Backtesting both assume this material.

// module · 02

Platform Coverage.

An honest survey of the six platforms most people actually use.

PAGES	PDF	WEB
14	/pdfs/platform-coverage.pdf	/learn/automation/platforms

What you'll learn

· Where each platform helps you and where it traps you

· What 'broker-of-record' really commits you to

· Latency, fees, and data-quality differences that move PnL

· Migration paths between platforms without losing history

· When to roll your own and when not to

Prerequisites

None — readable alongside Coding Fundamentals.

Leads into

Per-platform deep dives, then Data Engineering.

// module · 03

Backtesting.

How to design a backtest that does not lie to you.

PAGES	PDF	WEB
36	/pdfs/backtesting.pdf	/learn/automation/backtesting

What you'll learn

· Walk-forward, purged k-fold, and combinatorial CV

· Look-ahead leaks: the seven varieties and how to catch them

· Survivorship, point-in-time data, and corporate actions

· Slippage and fee models that match your venue

· When a backtest is 'good enough' to paper-trade

Prerequisites

Coding Fundamentals; Data Engineering recommended.

Leads into

Machine Learning (feature design) and Operations (deployment gates).

// module · 04

Data Engineering.

Bars, ticks, vendors, alignment, and the boring layer everything else rides on.

PAGES	PDF	WEB
28	/pdfs/data-engineering.pdf	/learn/automation/data-engineering

What you'll learn

· Bar construction: time, tick, volume, dollar, and imbalance

· Vendor comparison: who covers what, at what cost

· Point-in-time joins and the as-of pattern

· Schema design for a research warehouse

· Backfills, replays, and gap detection

Prerequisites

Coding Fundamentals.

Leads into

Backtesting and Machine Learning both consume this output.

// module · 05

Machine Learning.

Models that survive contact with markets, and how to tell which do not.

PAGES	PDF	WEB
40	/pdfs/machine-learning.pdf	/learn/automation/machine-learning

What you'll learn

· Feature engineering for non-stationary data

· Model comparison: linear, tree, gradient-boost, neural

· Cross-validation that respects time and embargo

· Overfitting: detection, diagnosis, and prevention

· The 15 production gates a model must clear before going live

Prerequisites

Backtesting and Data Engineering.

Leads into

Operations — once a model is live, the desk owns it.

// module · 06

Operations, CLI & Recovery.

The desk is the system. Ten procedures, six incidents, six trees, twelve monitors, twenty gates.

PAGES	PDF	WEB
32	/pdfs/operations.pdf	/learn/automation/operations

What you'll learn

· Runbook procedures: deploys, rollbacks, kill-switches, drills

· Incident response: detection → containment → postmortem

· Recovery decision trees for the six failures that bite first

· Monitoring across infra, data, model, and PnL layers

· Cadence gates: pre-launch, daily, weekly, monthly, quarterly

Prerequisites

All prior modules — operations assumes the system exists.

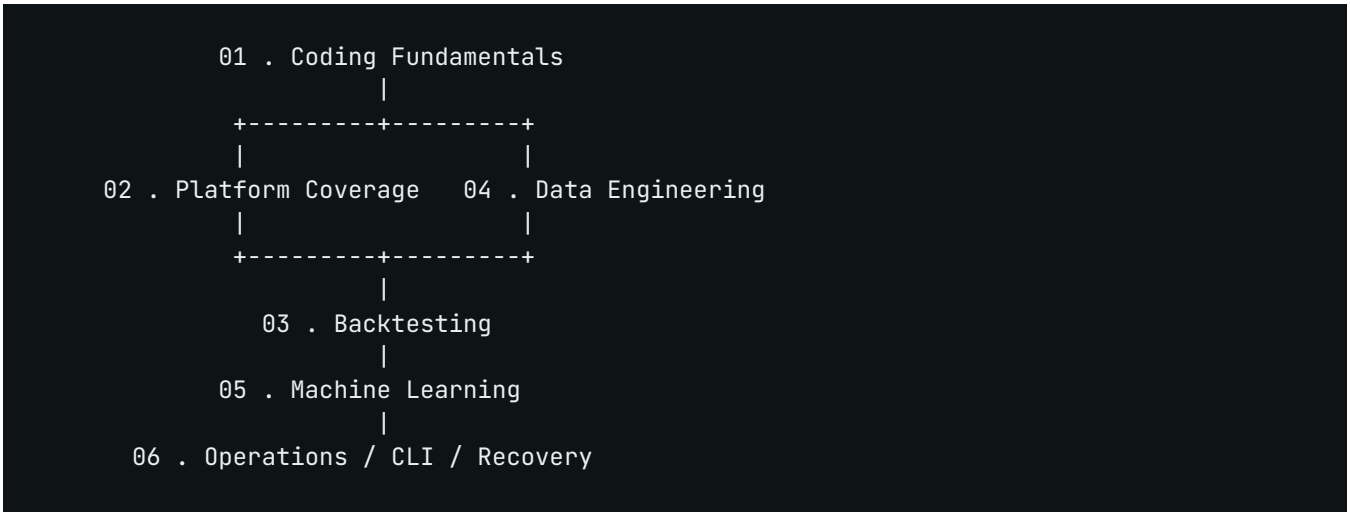
Leads into

Closes the curriculum.

// dependency map

How the modules connect.

Arrows mean 'the downstream module assumes you read this'. Boxes in parentheses are optional but make the next module easier.



Modules 02 and 04 are siblings — read them in either order, but before 03. Module 03 (Backtesting) is the linchpin: if you skip it, every later module degrades.

// study guide

How to use this material.

Read with a terminal open.

Every code block is meant to be typed, not copy-pasted. Typing forces you to read it once before running it. Copy-paste skips that step and most bugs hide in that gap.

Keep one repo for the whole curriculum.

Create one private repo, commit after every module, and keep your notes in /docs alongside the code. The git history is the best study journal.

Do the exercises in paper-trading mode first.

Every strategy in the curriculum is meant to be run against a paper account for at least one trading week before any real money. The Operations module makes this a hard gate.

Argue with the material.

Where we take a side and you disagree, write your counter-argument in /docs. The fastest way to learn this material is to find the place where it is wrong for your situation.

Re-read 06 every quarter.

Operations is the only module designed to be re-read. The twenty-gate checklist is meant to be run, not memorised.

// honest limits

What this curriculum is not.

It is not a get-rich-quick path.

Nothing in the 181 pages tells you which symbols to trade or which strategy will work this year. The whole point is to teach you how to find that out for yourself, honestly.

It is not an HFT curriculum.

Anything sub-second is out of scope. The math, the data, and the infrastructure are different enough that mixing the two would weaken both.

It is not options-specific.

The principles transfer, but greeks, expiry mechanics, and implied-volatility surfaces are not covered. Options need their own curriculum.

It is not financial advice.

Educational only. Past performance does not guarantee future results. We do not know your tax situation, your risk tolerance, or your capital base. Decisions are yours.

It is not a substitute for a postmortem.

Reading this will not save you from your first live blow-up. Running it through Module 06's incident playbook will help you survive the second one.

// glossary

Recurring terms.

TERM	MEANING
As-of join	A point-in-time join that uses the latest record available at or before the timestamp on the left side. The default join for any feature-vs-bar lookup.
Embargo	A gap between training and validation folds, used in purged k-fold CV to prevent leakage through auto-correlated labels.
Kill-switch	A physical or endpoint control that flattens all positions and blocks new orders within 60 seconds. Drill it quarterly.
Look-ahead leak	A backtest defect where the model sees data it could not have seen at the moment of decision. The single most common reason a backtest lies.
Model hash	A content hash of the model artifact, used as the identity check at deploy time. Live /healthz must report the expected hash.
PIT	Point-in-time. A dataset where every cell reflects only information available at that cell's timestamp.
PSI	Population Stability Index. A drift metric between training and live feature distributions. Trigger at 0.25.
Purged k-fold	A cross-validation variant where overlapping label windows are removed from the training fold to prevent leakage.
Shadow mode	Running a new model in production reading live data but not placing orders, to compare predictions before promoting capital.
Slippage	The difference between the modelled fill price and the realised one. Always model it, even when it is small.
Walk-forward	Training on a rolling historical window and evaluating on the next-step window, advancing one step at a time.

// further reading

Sources and further reading.

Each module ships its own source list. These are the texts the curriculum returns to most often, across modules.

[1] López de Prado — Advances in Financial Machine Learning.

https://www.wiley.com/en-us/Advances+in+Financial+Machine+Learning-p-9781119482086

[2] Bailey & López de Prado — The Probability of Backtest Overfitting.

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2326253

[3] Google SRE Book — incident response and postmortems.

https://sre.google/sre-book/table-of-contents/

[4] Twelve-Factor App — config and backing services.

https://12factor.net/

[5] SEC Rule 15c3-5 — Market Access Rule.

https://www.sec.gov/rules/final/2010/34-63241.pdf

[6] CFTC Regulation Automated Trading (Reg AT).

https://www.cftc.gov/sites/default/files/idc/groups/public/@lrfederalregister/documents/file/2015-29842a.pdf