Systems, Failure, and Growth: An Engineer's Journey in Business Architecture

Anthony Singgih, M.Eng. (Mechatronics)

University of Melbourne, November 2025

1 Introduction

I have always been interested in how businesses actually operate—how decisions turn into movement, money, or failure. But the way people describe it never made much sense to me. I come from engineering, where systems behave according to rules, and every output has a measurable input. When I tried to study business, I found a language full of vague terms and undefined equations. It was like entering a field that refused to quantify itself. So I treated it as an engineering problem: if the logic isn't explicit, extract it. I started mapping businesses the way I would map a machine—identifying inputs, processes, outputs, and feedback paths until the behavior made sense. This is a conceptual paper, intended more to showcase the breadth of my understanding as a self-described generalist than to demonstrate technical depth. This is the early work of someone trying to become a generalist systems thinker.

2 Abstract

This paper presents a systems-oriented interpretation of business architecture from the perspective of an engineer trained in mechatronics. Rather than treating business as a collection of abstract managerial concepts, the work reframes it as an integrated, layered system analogous to engineered processes composed of inputs, transformations, feedback loops, and outputs. A five-layer conceptual model is introduced—spanning Operations, Analytics, Engineering, Management, and Investment—to provide a simplified yet structured framework for understanding how data, decisions, and value propagate through an organization. The paper emphasizes the primacy of operational systems as the origin point of all business data, arguing that accounting, marketing analytics, and data science function primarily as interpretive layers built atop transactional reality. Through illustrative examples—including Point-of-Sale (POS) systems, Material Requirements Planning (MRP), and Enterprise Resource Planning (ERP) architectures—the paper traces the flow of information from customer interaction through production and analytics, highlighting the importance of system integration and data fidelity. Engineering concepts from control theory, embedded systems, and estimation theory are applied to demonstrate how business infrastructure increasingly resembles cyber-physical systems governed by feedback, uncertainty, and optimization. In its later chapters, the paper examines emerging technologies such as artificial intelligence, retrieval-augmented generation, and agent-based system architectures as extensions of traditional analytics, and situates leadership decision-making within a systems framework rather than purely managerial abstraction. Finally, the work reflects on professional development, arguing that modern technical careers require strategic positioning across systems layers rather than narrow specialization. While exploratory in nature, this paper aims to provide a unifying lens through which engineers, analysts, and business practitioners can better comprehend how operational execution, data interpretation, and strategic control interact within contemporary organizations.

3 Layers of Business Systems

Layer	Key Question
1. Investors & Banking	Should this business exist?
2. Management & Leadership	How to lead a business?
3. Analytics & Support Systems	How to interpret business information?
4. Operations	How to run a business?
5. Engineering	How to design, maintain or optimize a business machine, process or product?

I have encountered some models that try to capture how business systems work. Lean Six Sigma, for example, already describes efficiency and process control, but operates on a technical and statistical level that most people never touch. It deals with measurements, variances, and defect ratios—useful in manufacturing but difficult to visualize outside it. My model is not meant to replace those frameworks; it is meant to simplify the view. Reduces the larger system to visible layers that even a fresh graduate can understand: leadership sets direction, analytics reads the data, operations get the work done, and engineering keeps the machine running. It's built for clarity, not certification—a way to see how a business functions without getting lost in complexity.

To keep this paper concise, the first two layers will be set aside, as they fall outside my expertise nor do I have the experience. For context, my profile shows that I hold a master's degree in engineering and have experience in operations and administration, including my time working as a crew member at McDonald's. I will explain these concepts through simple projects I've built to showcase my portfolio and demonstrate my understanding and ability to execute.

4 Operations

I'll begin with the layer I'm most familiar with. Operations form the core of any business; without them, actions aren't executed, products aren't made, and sales never occur. Following my approach to understanding operations, we must first examine various existing systems that are developed and embedded within business operational processes. **Focus on how data moves throughout the business.**

I'll start with something simple—a kiosk or point-of-sale (POS) system. It's a necessary component because it captures every transaction at the front line of the business, turning customer actions into data that drive production, inventory, and financial decisions.

4.1 POS System

To illustrate how the system works, I'll vibe code a short, bare-minimum POS app, with the help of AI. The goal isn't to show programming skill, but to make the process easier to understand. I will code three main functions of the system. Repository: https://github.com/anthonysinggih-alt/business-systems.git

A. TRANSACTION INPUT

The system allows the user to initiate a new sale and input basic transaction details. Each item entry includes the product name or ID, unit price, and quantity. Items are stored temporarily in a cart until the sale is completed. The admin must first enter the item details into the system.

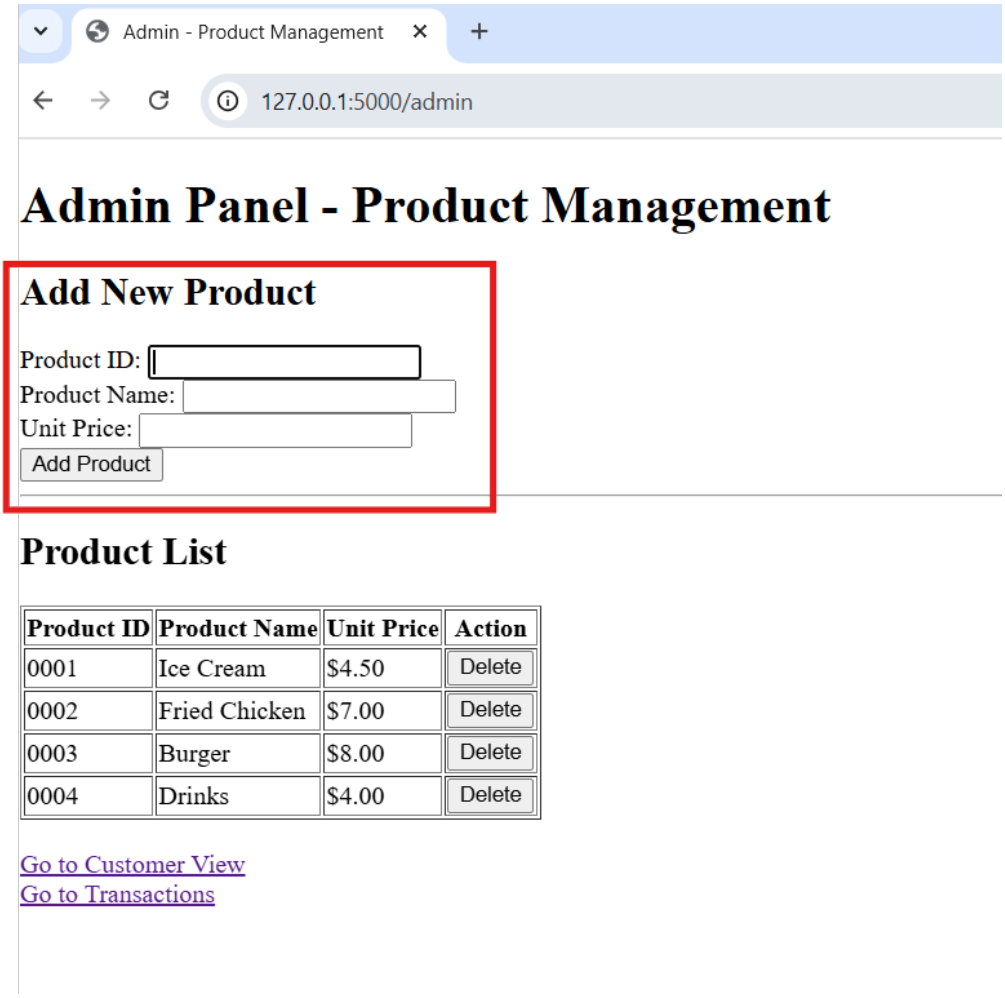


Figure 1: Admin interface for entering product details into the POS system.

B. PAYMENT PROCESSING

Once all items have been entered, the system calculates the subtotal, displays the cart items and sums the results. The system can optionally apply tax or service charges.

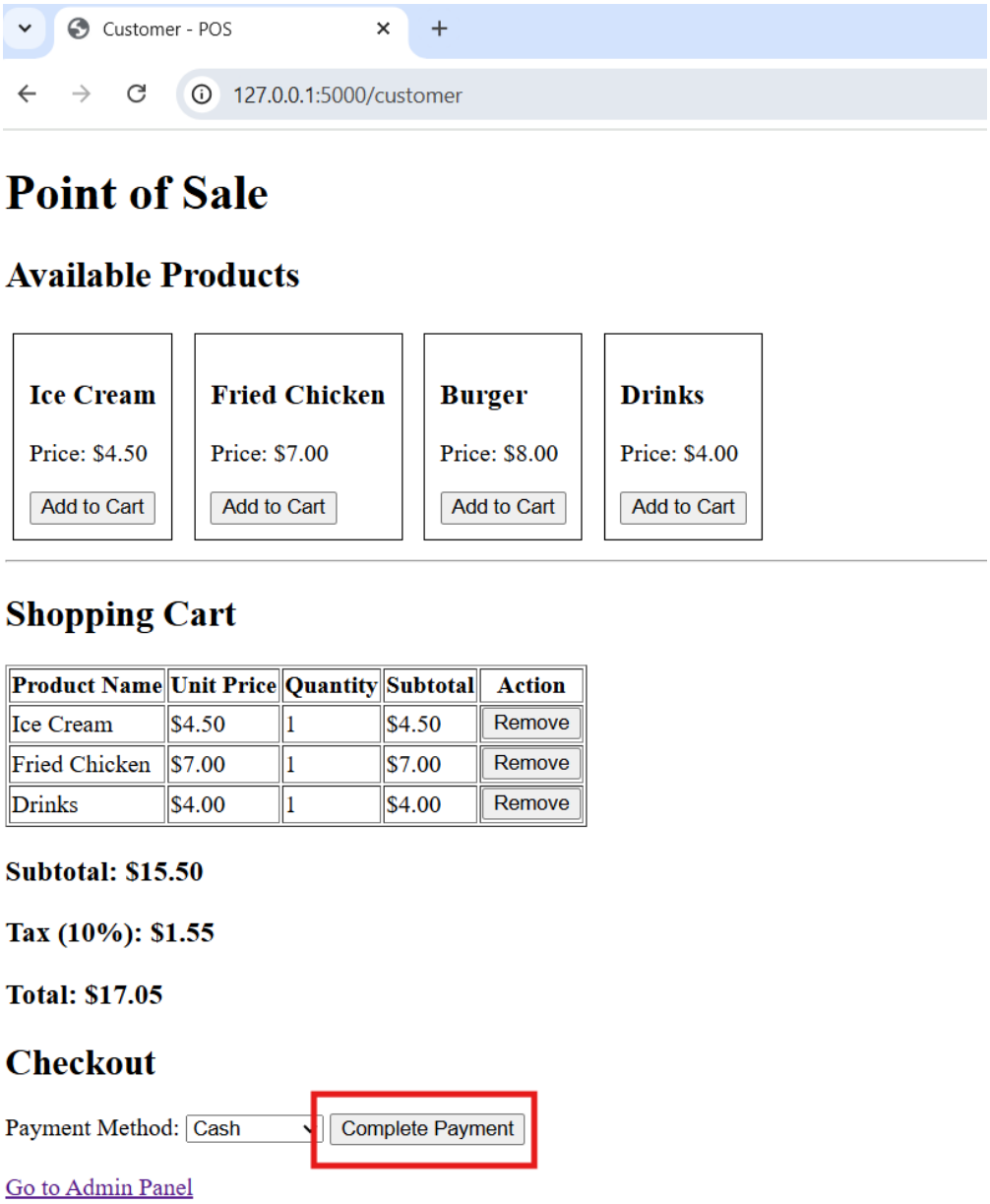


Figure 2: Customer interface showing shopping cart and payment options.

C. DATA STORAGE & ENTRY

After payment is confirmed, the system saves the completed transaction in a persistent storage format such as a CSV file. Each record includes the timestamp, list of items sold, subtotal, total amount, and payment details. Once stored, the system resets and becomes ready for the next transaction.

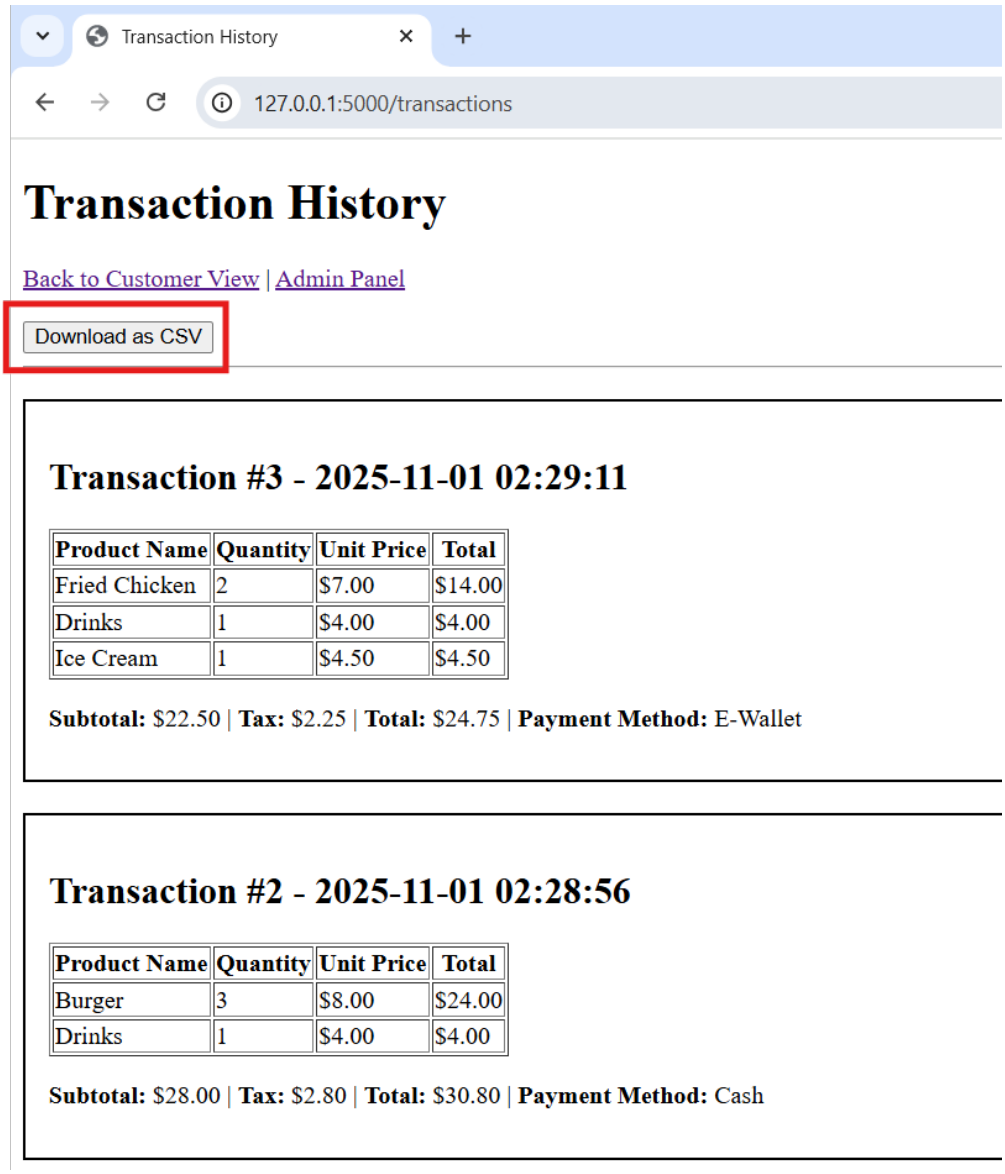


Figure 3: Record showing stored order data after transaction.

4.2 Software Architecture

The software described above is implemented using Flask and follows a REST architecture, which is a common framework for modern web applications. I reiterate that the purpose here is not to demonstrate software engineering proficiency, but to identify the origin and flow of data within business operations. **Understanding how data moves through a business system** is the central objective of this exercise.

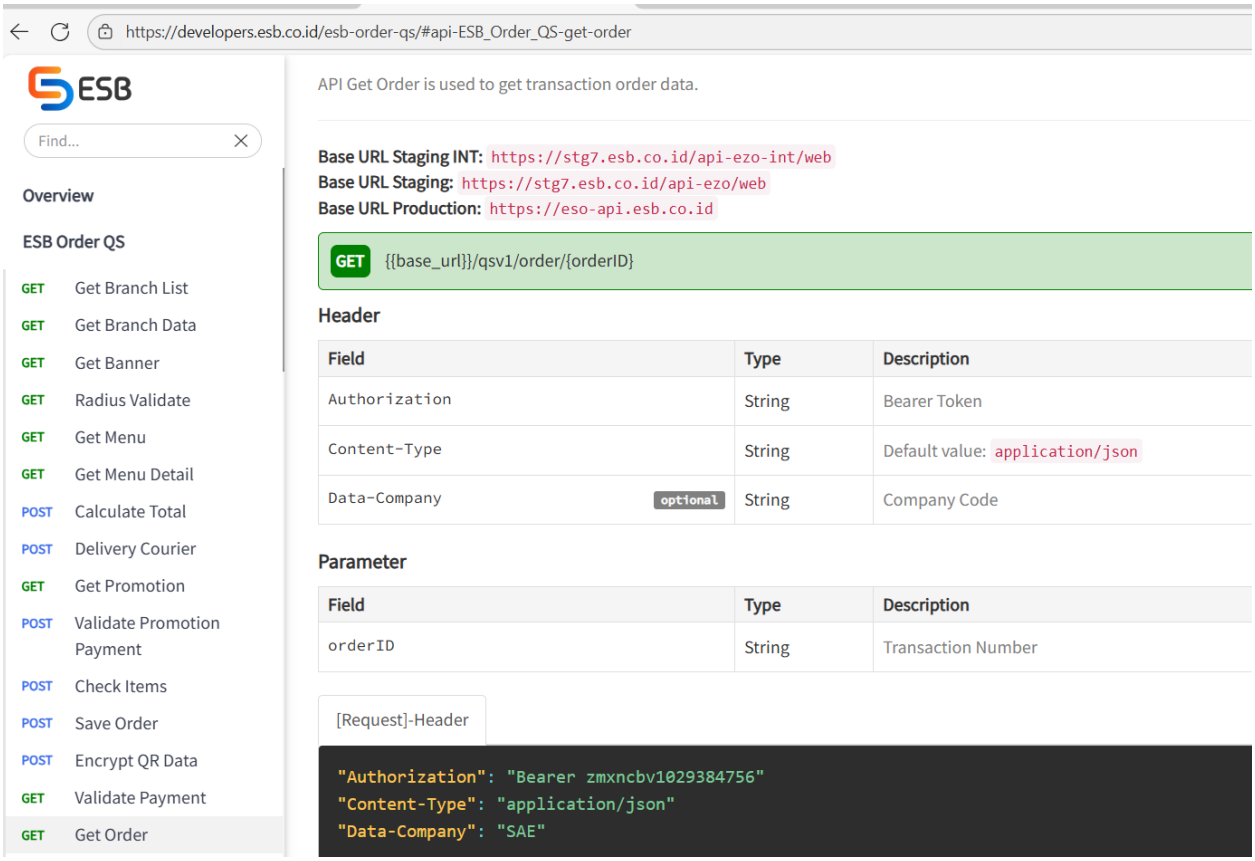


Figure 4: External API to extract order data.

Above is an example of a commercialized application that uses a similar architecture—the system currently used by my family's franchise business.(https://developers.esb.co.id/). As documented in its API reference, the software exposes endpoints that define how external systems can communicate with it. Reviewing such documentation is important for understanding how operational data is generated, transmitted, and accessed, even when the software is developed and managed by a third-party provider.

4.3 MRP System

I'll need a second system to serve as an example to support my point. Material Requirements Planning (MRP) is a system used to plan and control the materials and resources needed for production. It takes information from sales or production orders and calculates what raw materials, parts, and labor are required, as well as when they should be available. By doing this, MRP ensures that production has the right inputs at the right time—preventing shortages, reducing excess inventory, and keeping operations efficient. In essence, it translates customer demand into a clear, time-based plan for manufacturing or operations. I was responsible for administering such a system—MRPeasy (https://www.mrpeasy.com/resources/api/) for over a year in a construction manufacturing environment. My role involved maintaining accurate records of materials, procurement, production schedules, and invoicing. As with the POS, I will code three core functions of this system using the same architecture, for the purpose of examining and visualizing data flow. Repository: https://github.com/anthonysinggih-alt/business-systems.git

A. Bill of Materials (BOM)

A Bill of Materials (BOM) defines the hierarchical structure of products, listing all raw materials, components, sub-assemblies and even lead times required to manufacture a finished item. In Material Requirements Planning (MRP) systems, the BOM serves as a foundational element for calculating material needs, production schedules, and procurement plans.

The admin interface allows users to manage both base items—which can be procured—and component items—which can be manufactured using base items or other components. A recursive database function models these hierarchical relationships between components and their dependencies. All item data and relationships can be viewed and modified directly through the admin panel in the application.

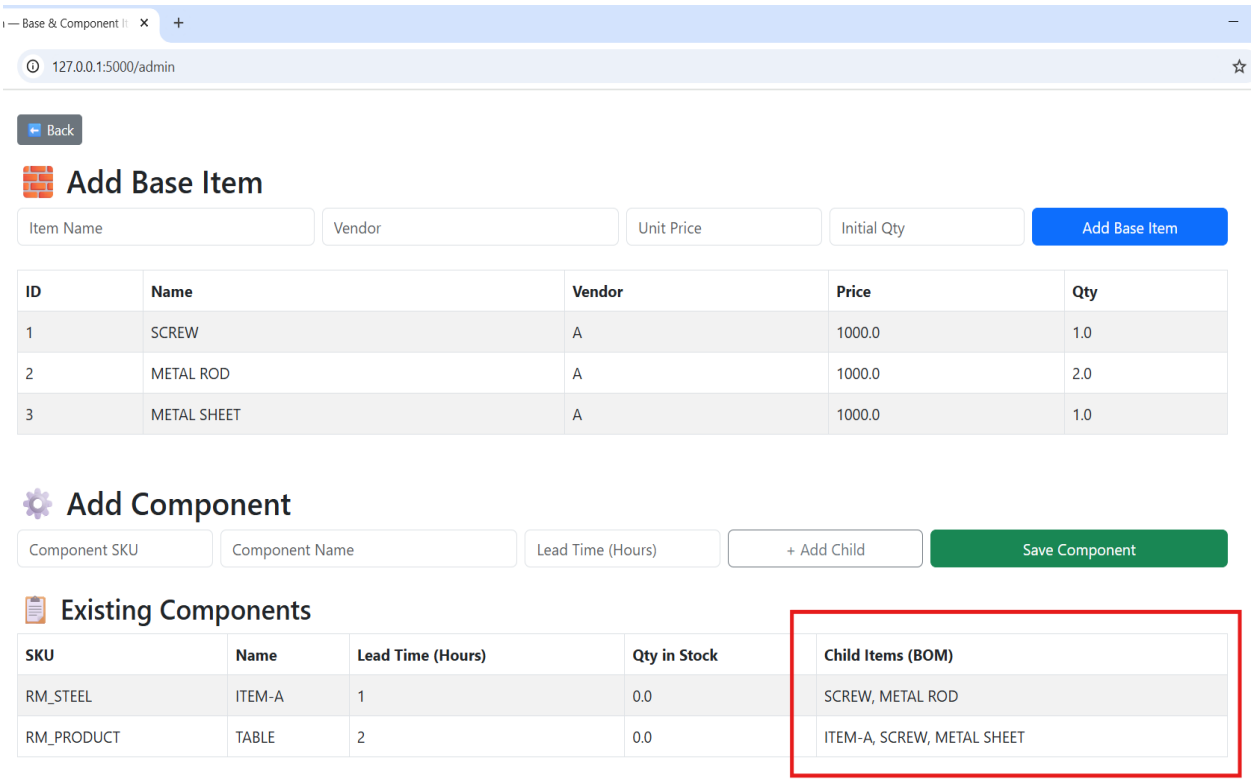


Figure 5: Admin interface for editing the Bill of Materials (BOM).

B. Procurement

This module handles the purchasing and replenishment of raw materials or base items.

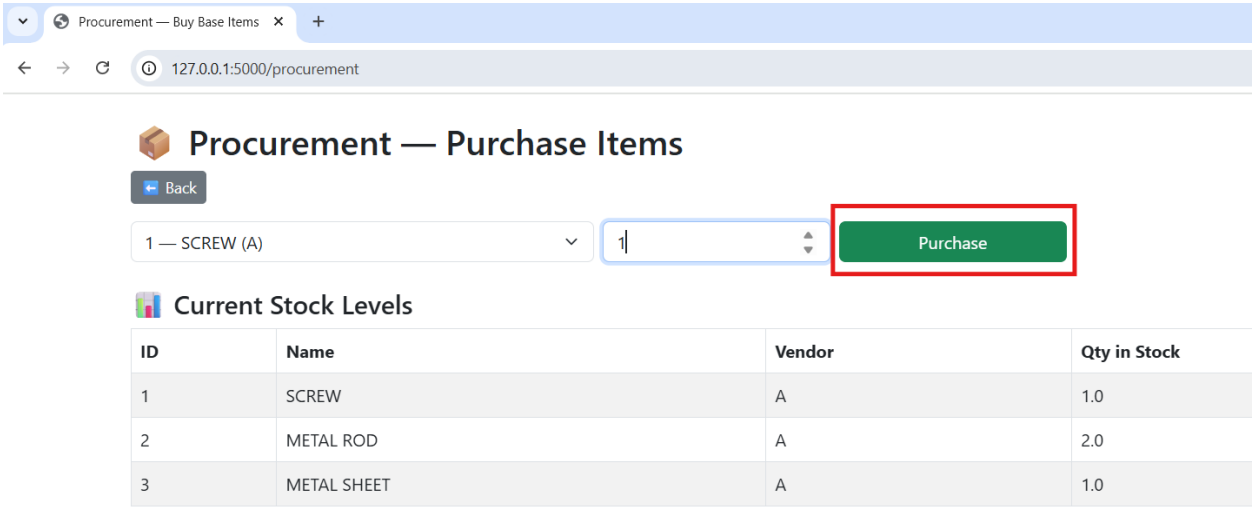


Figure 6: Procurement interface to add stock quantity.

C. Production Scheduling

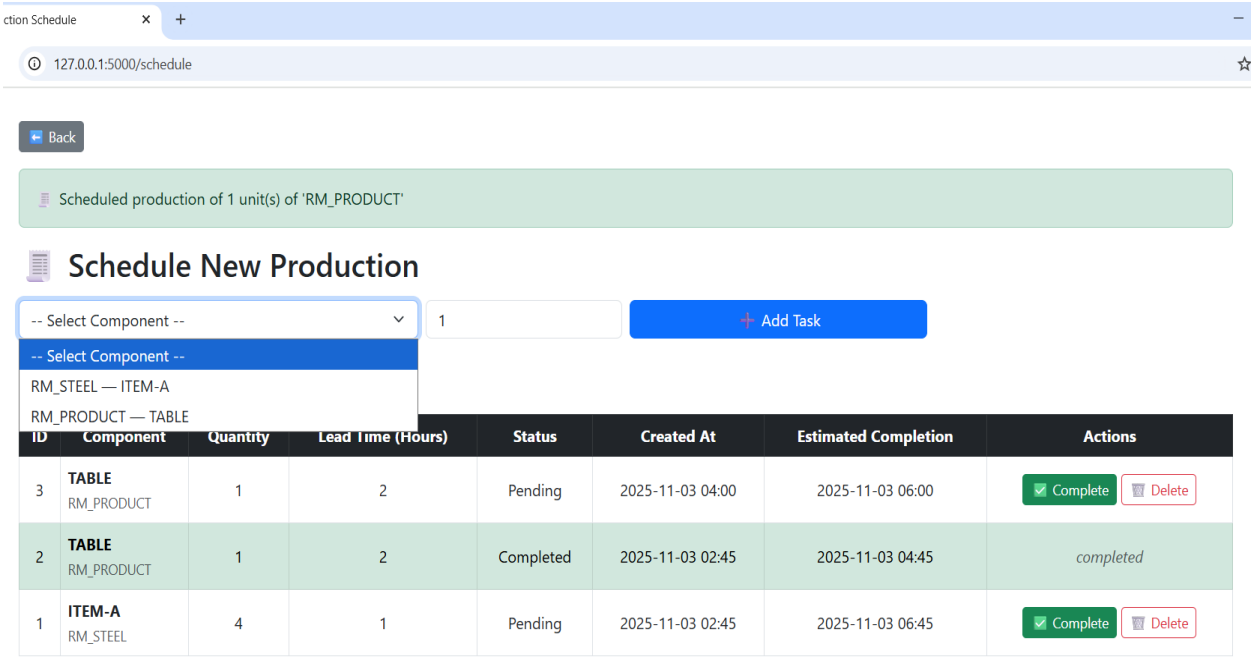


Figure 7: Task Scheduling

Ultimately, this enables us to create a scheduling function that calculates and displays the planned duration beneath the task list. In a full MRP application, there would also be a machine list that can be assigned to tasks, supporting both concurrent and sequential operations, which would then be visualized on a Gantt chart.

What is not shown here is that all **data originates from customer demand and product SOP.** Most MRP systems include features to record customer orders. This demand is then managed by an operator who handles procurement from suppliers and production planning shown above. The product SOP allows the operator to build a BOM by providing further detail—either referencing specific documents, such as engineering drawings, or clearly outlining the precise requirements needed to produce the product. These plans are then executed by operational staff performing the physical tasks. Throughout the process, all data are recorded in the system to allow resources and time to be used effectively by the organization.

4.4 Business Data

In today's world, nearly all operational data exists in the digital realm. While it is still possible to design similar systems using pen and paper, that approach is no longer practical in modern business environments.

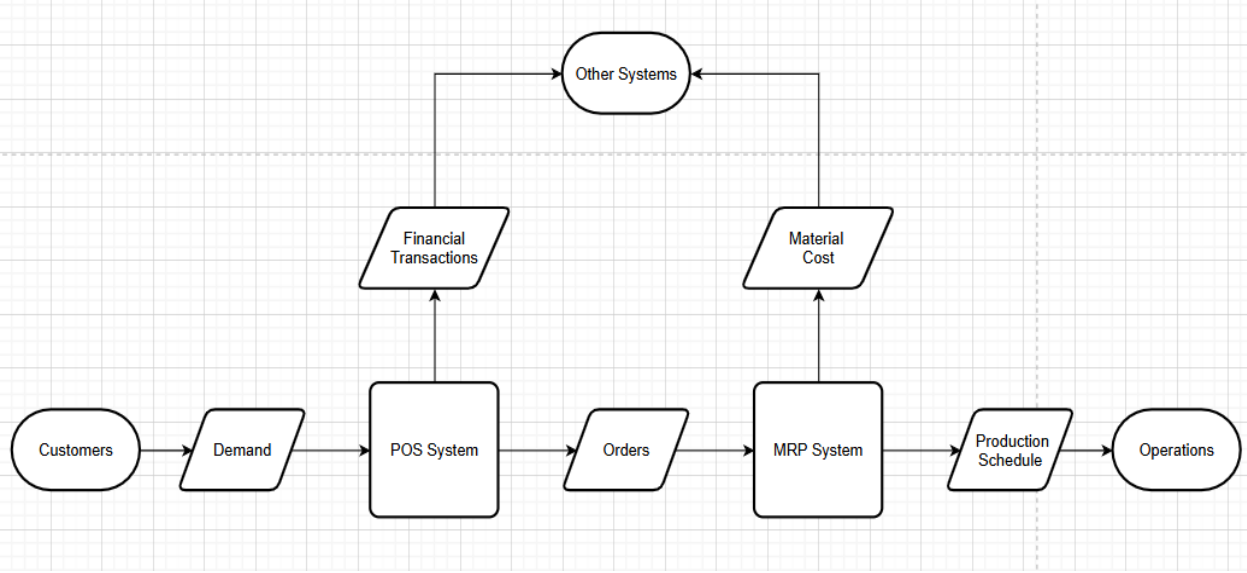


Figure 8: Flowchart showing potential interaction of systems.

The diagram above illustrates how these two systems can potentially interact, extending their functionality beyond isolated operations. This connection allows the POS to transmit purchase data to the MRP, which can then compute resource usage, time-based consumption, profit margins (excluding overhead), and short-term forecasts. Such integration enables data to inform analytics, automate repetitive processes, and improve coordination between departments. This entire attempt at visualization began with a simple question: **Where does the data actually come from?**

Today, complete software ecosystems exist to manage every aspect of a business, such as SAP. Ultimately, these integrated systems form what is known as Enterprise Resource Planning (ERP)—a centralized software platform that allows a business to track all of its core operations, from materials and production to sales, finance, and scheduling. Rather than running separate programs for each department, an ERP system consolidates them into one shared environment so that all functions operate on the same data. It's worth noting that despite the recent advancements in AI and software technology — and the variety of ERP options now available (such as Microsoft Dynamics, NetSuite, and Odoo) — the entry cost for these systems remains relatively high, particularly when it comes to implementation expenses.

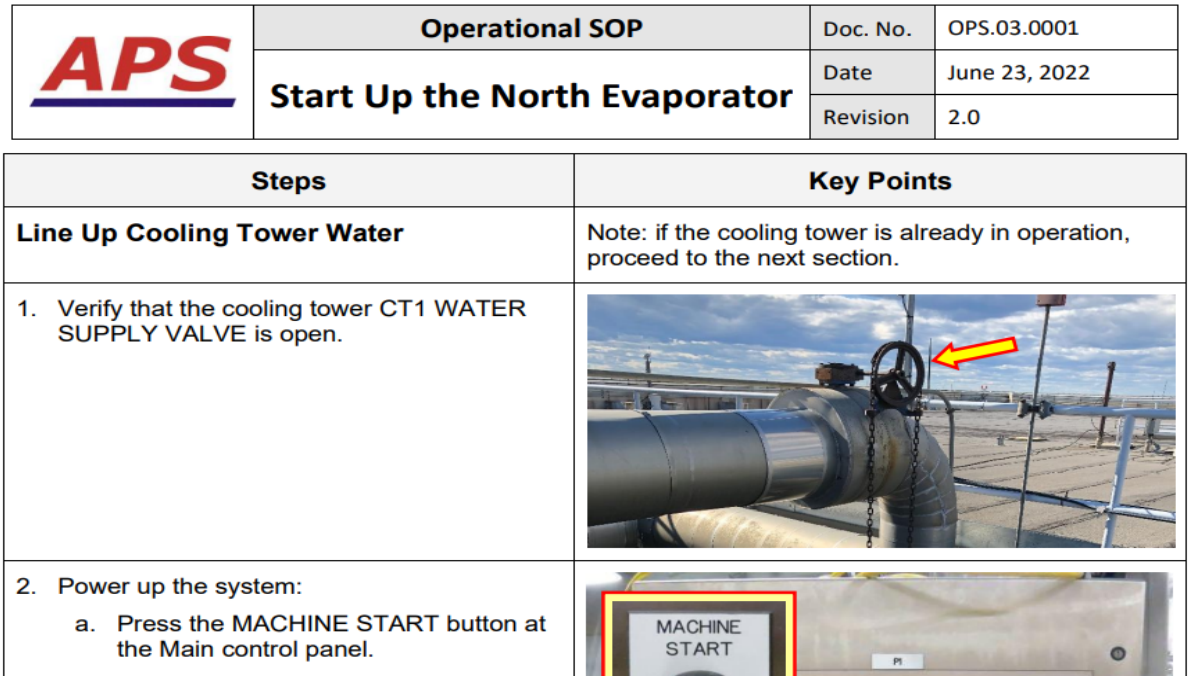
My family's enterprise developed its own software ecosystem in the mid 1990s by hiring fresh graduates and training them in coding. At that time, computers were not yet widely used, and their goal wasn't to showcase technical prowess but rather to overcome financial limitations that prevented them from purchasing commercial systems. The entire ERP system was built in legacy C++ and Java without overpowered AI.

4.5 Standard Operating Procedures (SOP)

An SOP (Standard Operating Procedure) provides clear, step-by-step guidance to ensure tasks are carried out consistently, safely, and in compliance with organizational or regulatory requirements, regardless of the operator. These procedures are often stored as PDF files scattered across various computer and devices within the company, unless there is a strict document control. There are many types of SOPs, but for simplicity, I will categorize them into two main types.

4.5.1 Product SOP

In my view, this represents the **most critical data within the entire organization.** Product SOPs consist of detailed work instructions and precise steps outlining how to manufacture the product or service. The story typically follows a familiar pattern: a genius identifies a gap in the market driven by customer demand and develops a creative or innovative solution. Through testing and refinement, this concept evolves from a prototype into a repeatable and standardized process capable of consistently producing a specific product or service for its intended purpose. That genius then realizes there's an opportunity to turn the idea into a profitable business and begins hiring people to perform the work. Soon enough, he discovers the need for a repeatable, well-documented procedure to ensure the product's quality and functionality can be consistently reproduced. This marks the birth of an original business—at least the idea of it. However, such ventures can still fail especially without thorough feasibility studies and proper financial planning. In other business models, such as franchising, the product SOPs are typically supplied by the franchiser.



APS	Operational SOP	Doc. No.	OPS.03.0001
	Start Up the North Evaporator	Date	June 23, 2022
		Revision	2.0

Steps	Key Points
Line Up Cooling Tower Water	Note: if the cooling tower is already in operation, proceed to the next section.
1. Verify that the cooling tower CT1 WATER SUPPLY VALVE is open.	
2. Power up the system: a. Press the MACHINE START button at the Main control panel.	

Figure 9: Sample SOP from https://aps-online.net/

For legal reasons, I obviously cannot disclose an actual company's SOP, as it would reveal proprietary methods of production. The example shown above is an operational SOP sample from a company specializing

in document control and SOP development. While a single SOP may not give an operator the complete picture of how the final product is made, it represents an essential piece of the overall process. When enough of these SOPs are compiled—from the beginning of production to the final stage—it becomes relatively easy to infer the entire manufacturing process. As I've said before, such documents are incredibly valuable. Take the Coca-Cola formula, for instance—worth billions of dollars and guarded with extreme secrecy. It's likely that even their full production SOP omits the exact list of ingredients. In most cases, however, SOPs do specify such details unless the product is so valuable that it is impossible to replicate elsewhere. I have seen them during my tenure with several different companies.

4.5.2 Supporting Documents

Although not as critical as the latter, this category includes various forms of documentation, such as training materials for on boarding new employees to ensure they can perform tasks correctly, guidelines for maintaining records of performance or activity outcomes, and business or workflow instructions on how to manage administrative data entry within the company.

All SOPs are continuously refined throughout a company's lifecycle. When an organization adheres to proper Quality Management System (QMS) or Good Manufacturing Practice (GMP) principles—such as those outlined in ISO 9001 (https://www.iso.org/standard/62085.html) or HACCP (https://www.fda.gov/food) for food products, it establishes processes to ensure that all documents are properly stored, controlled, reviewed, and updated. This systematic approach helps maintain consistent product quality, regulatory compliance, and overall organizational performance.

4.6 Other Operations Systems.

I had the opportunity to visit a steel distributor in Australia, where I observed impressive systems in action. The team was actively coordinating complex logistics, and a large display in the center of the office tracked their key performance metric — Delivery In Full, On Time (DIFOT).

These honorable mention systems — TMS (Transportation Management System) and WMS (Warehouse Management System) — are likely critical to their operations. The TMS helps plan, execute, and optimize the movement of goods, ensuring efficient transport routes and cost control, while the WMS manages inventory, storage, and order fulfillment within the warehouse. Together, they enable real-time visibility and coordination across logistics and supply chain processes.

4.6.1 Digital Twin

A good example of a complex digital system that is applicable to operations — one that challenges even engineers at their level of problem-solving — would be digital twins, though I won't go into detail here.

"Often times, we are limited to sparse, indirect and noisy observations to try and infer what's going on." - Karen Willcox (2023), Ph.D, MIT.

When I run into a subject that's beyond my depth, I go back to what university taught me best — find someone smarter, learn from them, and give them credit. A digital twin is a virtual representation of a physical object or system that uses real-time data to accurately reflect its real-world counterpart's behavior, performance and conditions (IBM, 2025). It's fair to say that my graduate studies were what first introduced me to this concept. The simplest way to explain this to an average person, without getting into the complexity of sensor systems or state observers, is as follows:

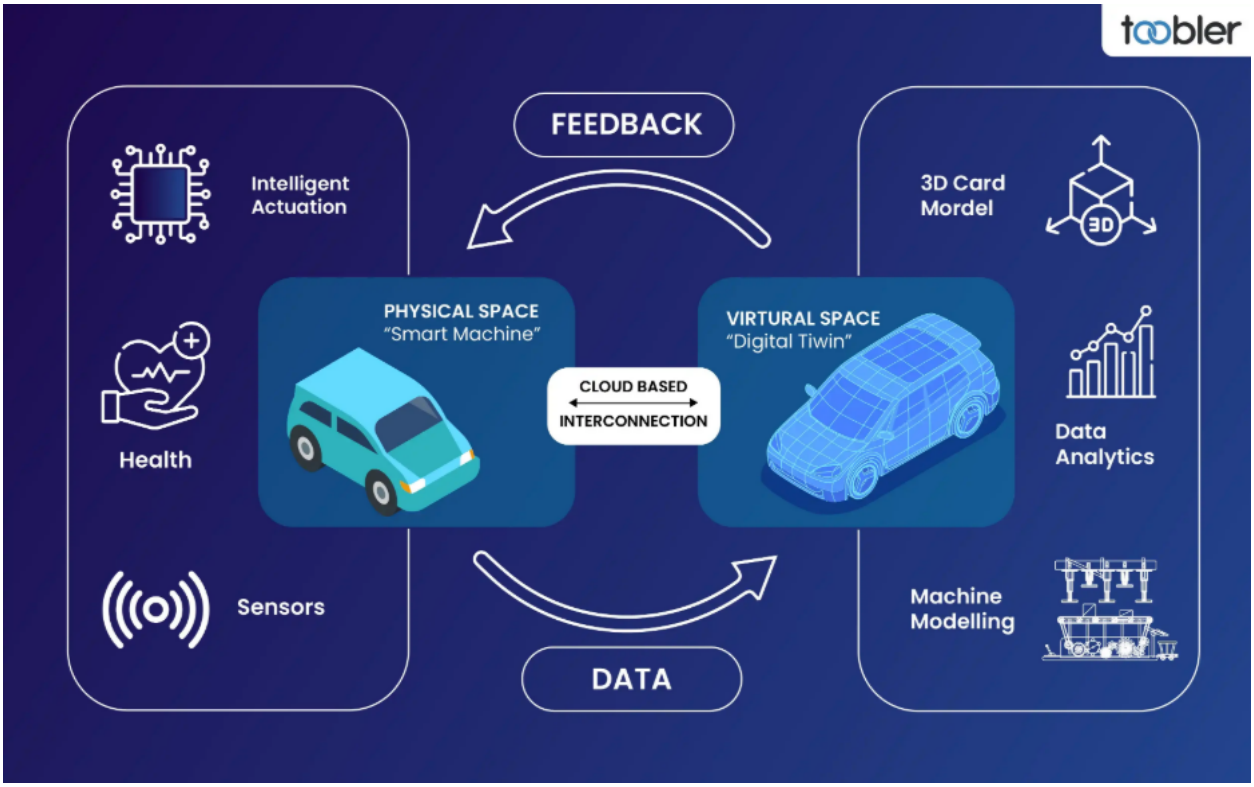


Figure 10: Sample Image from https://www.toobler.com/

A doctor can't always tell what's wrong with a person without looking inside their body, just as we can't always tell what's wrong with an airplane without taking it apart. Sometimes, we have to rely on other observations that are connected to the real problem. That is why we have a digital twin to play with. More about this in *Section 6.4 Sensor Systems*.

4.7 Human Resources & Communication

All of these systems exist to support the core purpose of business operations: enabling people to work efficiently together. Traditional management operations extends beyond data and automation—it depends on communication, coordination, and human decision-making.

What I present here is not a management framework, but rather the viewpoint of an engineer or systems analyst seeking to understand how these components interact to form what I refer to as business systems, i.e the title of this paper.

I'd like to emphasize that having competent "human" management—people who truly understand human behavior, resources and emotions—is non-negotiable in any operation. There are simply too many factors that cannot be controlled or fully described, no matter how complex or advanced our systems and AI become, especially when dealing with situations that may involve sensitive information or events.

5 Analytics & Support Systems

This is the point where business systems shift from execution to interpretation. I don't come from a professional analytics background, but I've worked closely with live operational systems and have seen how the data they generate becomes the foundation for business insights. My graduate coursework in Machine Learning and an IBM AI certificate give me a conceptual grasp of analytical techniques, even if I'm not yet applying them at specialist depth. In the previous chapter, I described how operational systems produce data—POS transactions, MRP runs, inventory movements, and financial events. Now I want to examine what happens afterward: how that operational record turns into the reports, forecasts, and analyses that shape business decisions.

The guiding question is simple: **Where does the data come from?** Always in operations. Every financial metric, sales projection, or customer analysis ultimately traces back to an operational event captured somewhere in the system landscape. This suggests a useful mental model: Operations generate data; Analytics interprets it. The latter cannot function without the former, just as accounting cannot produce financial statements without underlying transactions.

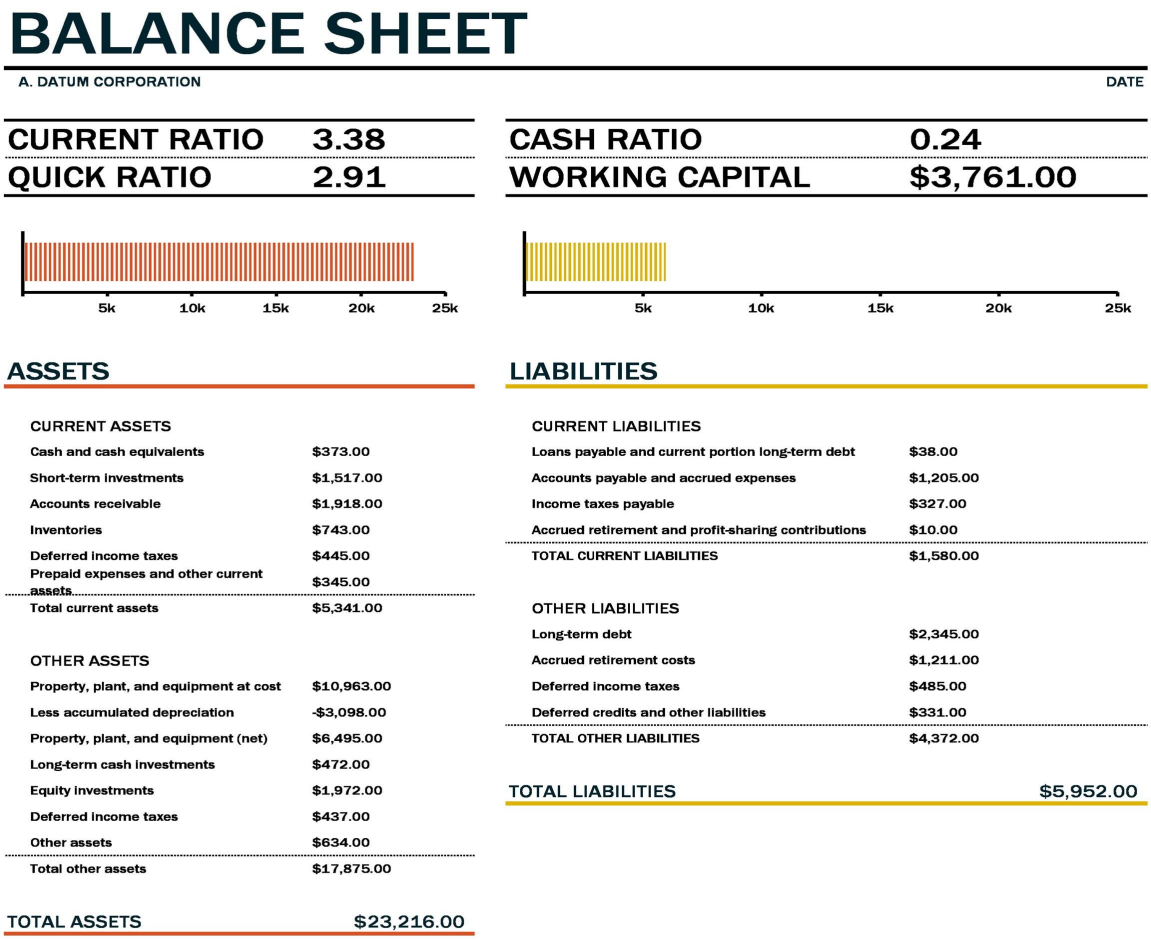
In this section, I'll look at several analytical domains—accounting, marketing analytics, and data science—not as an expert, but as someone mapping how they depend on and draw from the operational systems I've worked with. The emphasis remains on data origins and flow rather than technical depth, consistent with my generalist perspective.

5.1 Accounting

In my humble opinion, with only minimal formal learning in this subject at the undergraduate level, I consider **accounting to be the most critical supporting function within any business system.** Accounting may be formally defined as "the process of identifying, measuring, recording, and communicating economic information to permit informed judgments and decisions by users of the information" (American Accounting Association, 1966). Both my father and grandfather have consistently emphasized the central role of accounting in managing their enterprise since I was a child, applying this principle as a foundational methodology.

Every transaction between the business and its customers, every procurement decision made by staff, each payroll entry, and even the depreciation of assets or the loss of tools occur at the operational level — and all must be recorded. Without these underlying events taking place in operations — whether a product is produced, sold, or a service is delivered — there would be no data points to capture. Consequently, the balance sheet, income statement, and other financial records would simply reflect zero. At some point in history, an ingenious mind conceived the double-entry accounting system — a method designed to ensure accuracy and internal consistency in financial records. It functions as an inherent self-verification mechanism, applicable both through traditional means such as paper ledgers and in modern digital systems. Building upon this foundation, fundamental equations such as Assets minus Liabilities equals Equity, along with financial ratios that measure debt, liquidity, and performance, allow businesses to interpret the meaning behind numerical values. While these calculations require no calculus, they yield insights that are both practical and essential to informed decision-making. All I am emphasizing is that this is simply a derivative, **an observation of operational data over time for every financial transaction.**

While accounting is grounded in operational activity, many elements of financial reporting—such as depreciation, provisions, revenue recognition, inventory valuation and even goodwill—require judgment, estimation, and policy choices that go beyond simply recording raw events.



BALANCE SHEET

A. DATUM CORPORATION — DATE

CURRENT RATIO	3.38	CASH RATIO	0.24
QUICK RATIO	2.91	WORKING CAPITAL	$3,761.00

ASSETS

CURRENT ASSETS	
Cash and cash equivalents	$373.00
Short-term investments	$1,517.00
Accounts receivable	$1,918.00
Inventories	$743.00
Deferred income taxes	$445.00
Prepaid expenses and other current assets	$345.00
Total current assets	$5,341.00

OTHER ASSETS	
Property, plant, and equipment at cost	$10,963.00
Less accumulated depreciation	-$3,098.00
Property, plant, and equipment (net)	$6,495.00
Long-term cash investments	$472.00
Equity investments	$1,972.00
Deferred income taxes	$437.00
Other assets	$634.00
Total other assets	$17,875.00

TOTAL ASSETS $23,216.00

LIABILITIES

CURRENT LIABILITIES	
Loans payable and current portion long-term debt	$38.00
Accounts payable and accrued expenses	$1,205.00
Income taxes payable	$327.00
Accrued retirement and profit-sharing contributions	$10.00
TOTAL CURRENT LIABILITIES	$1,580.00

OTHER LIABILITIES	
Long-term debt	$2,345.00
Accrued retirement costs	$1,211.00
Deferred income taxes	$485.00
Deferred credits and other liabilities	$331.00
TOTAL OTHER LIABILITIES	$4,372.00

TOTAL LIABILITIES $5,952.00

Figure 11: Balance Sheet Template from https://templatelab.com/

I will not delve into the technical mechanics of debit and credit notes, ledger maintenance, or bookkeeping processes, nor into the auditing procedures employed to detect irregular transactions or perform reconciliations. This is, after all, not an accounting paper. Suffice it to say that the information presented in the hypothetical balance sheet template above is sufficient to indicate to upper management whether the ship is sinking or capable of assuming additional risk. Before my former professors begin lecturing me on liquidity ratios, debt structures, or the finer points of cash flow, let me acknowledge that there are indeed statements designed for those purposes. The cash flow statement, for instance, reflects the movement of liquid funds—money flowing in from sales and out through payroll or supplier payments. Likewise, the income statement provides a view of profitability, expense allocation, and fiscal performance.

These instruments collectively offer a sufficient foundation for understanding the essentials of accounting without venturing into the intricate arguments often held among accounting professionals over specific taxation rules, depreciation classifications, or regulatory interpretations. It is worth noting, however, that such debates are precisely what enable professional consultants to advise both governments on tax policy and corporations on how best to position their strategies and capital structures within those same regulatory frameworks, at the same time. (the joke—no, this is not a guide to minimizing taxes, so please do not call the ATO or PwC.)

5.2 Marketing

I recall taking an introductory course in this subject during my undergraduate studies at the persuasion of my ex-girlfriend. Much like accounting, without any transactional data between the business and its customers for every sales that occur, where a product or service gets delivered, and the information is recorded within a Customer Relationship Management (CRM) system, we do not have the data to report on anything. **This is also just a derivative of the observation of the operational data.** Technically speaking, I consider sales a component of operations, particularly in the traditional sense where it entails allocating human and logistical resources to carry out repeatable activities such as prospecting, pitching, and closing deals.

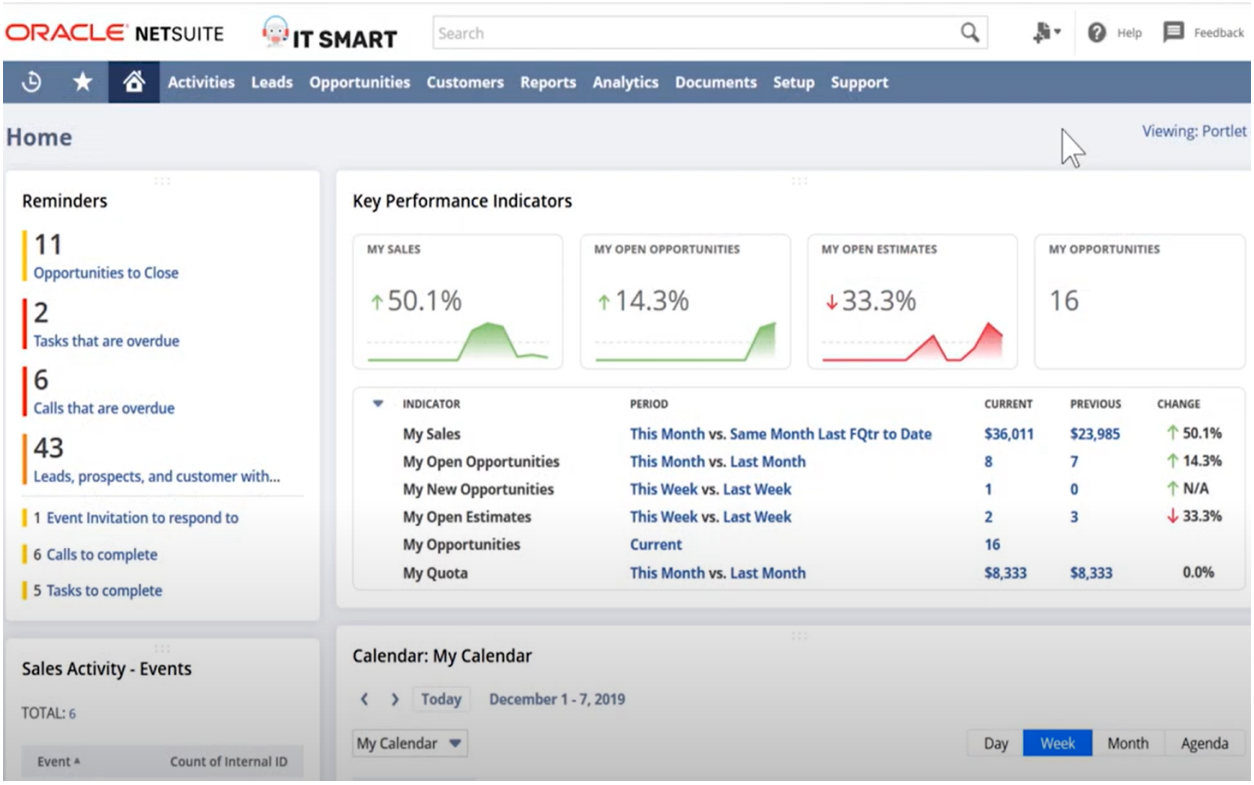


Figure 12: NetSuite CRM Dashboard

To clarify, sales and marketing are not the same thing. However, since I didn't go through the process of tracing real data from a CRM—as I did with other systems—I'll have to describe it conceptually. Marketing, at first glance, presents itself as a broad and abstract subject. Much like accounting or other commerce disciplines, it introduces you to questionably excessive and pretentious vocabularies. You begin with frameworks such as the marketing mix and the product lifecycle, followed by topics like customer behavior and market segmentation. Soon after, the coursework shifts toward case studies—analyzing brand value and asking why one company manages to sell more effectively than another, or why its sales process runs more smoothly.

When I first encountered this at eighteen, I was completely lost. My initial paper left me wondering: **where exactly does this data come from?** Not just figures copied from the internet or a Wikipedia page claiming Coca-Cola sold a certain number of products that year. I eventually realized that much of the exercise was about finding verifiable data and citing it properly, rather than about understanding how concepts like brand equity or trust value translate into sales performance when products hit the shelves. Of course, even at that age, I understood that just because something didn't make sense to me didn't mean it didn't work—or that it was useless. At the very least, I reasoned that whether it operates through psychology or some magic to influence customers, it still works. In the end, it became more about creative writing.

Obviously, my view has indeed changed as I've matured and experimented with those theories myself.

5.2.1 Robotic Process Automation (RPA)

This presents an excellent opportunity to illustrate how RPA and data-analytics (such as in the UiPath training dataset) can support marketing-operations alignment. By comparing product performance across regions, channels, and customer segments, marketing teams can identify which products are in demand where — and just as importantly, which are under-performing or redundant. Those insights enable operations to allocate MRP resources more effectively (e.g., ordering from specific suppliers, adjusting production runs) and provide stakeholders with evidence-based input on whether to continue, phase out or invest in particular machines, processes or product lines. Presenting these findings in a clear and decision-ready format helps leaders maintain tighter control over sales, resource deployment and cost-benefit trade-offs.

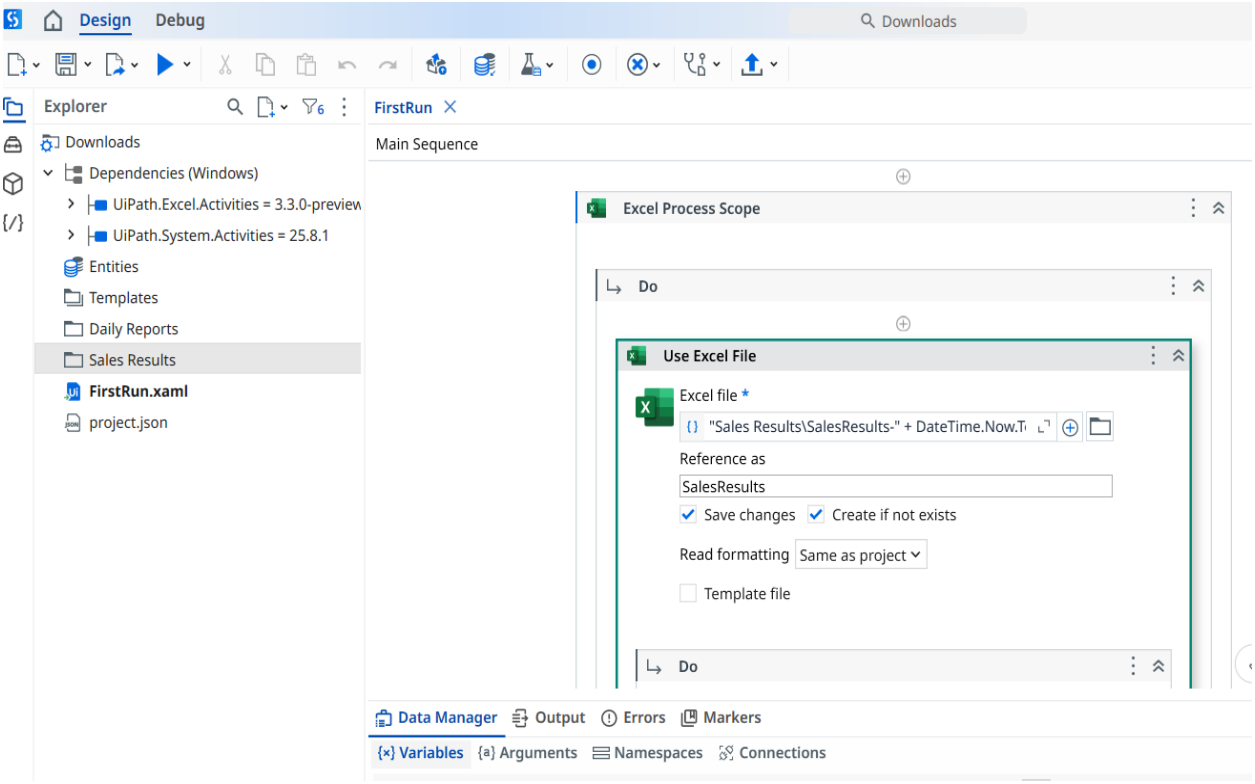


Figure 13: UiPath User Interface

RPA (Robotic Process Automation) is used to automate repetitive, rule-based tasks on a computer—such as processing Excel files, generating charts, or creating customized reports. In this training example, the dataset contains different coffee products sold by a business. RPA can automatically open each spreadsheet, summarize the raw data, and build a pivot table from it. Because the spreadsheets share the same structure, the automation can repeat the process across many files consistently and efficiently.

After completing my first UiPath academy exercise, I quickly realized why RPA exists—and why it felt unnatural to me. The task involved dragging and dropping activities to combine several Excel spreadsheets into a pivot table, yet the entire workflow was slower and more restrictive than simply writing a short Python script. That experience clarified that RPA is not designed for engineers or data-literate users; it exists for situations where systems have no APIs, no database access, and only a fragile graphical interface—typically legacy ERP screens, government portals, or proprietary industry software. In such environments, Python automation becomes unreliable, while RPA provides stable UI-level interaction through selectors, OCR, orchestrators, and even compliance logs. In short, RPA is essential when you cannot reach the backend at all—but whenever clean data access is available, code will always outperform it.

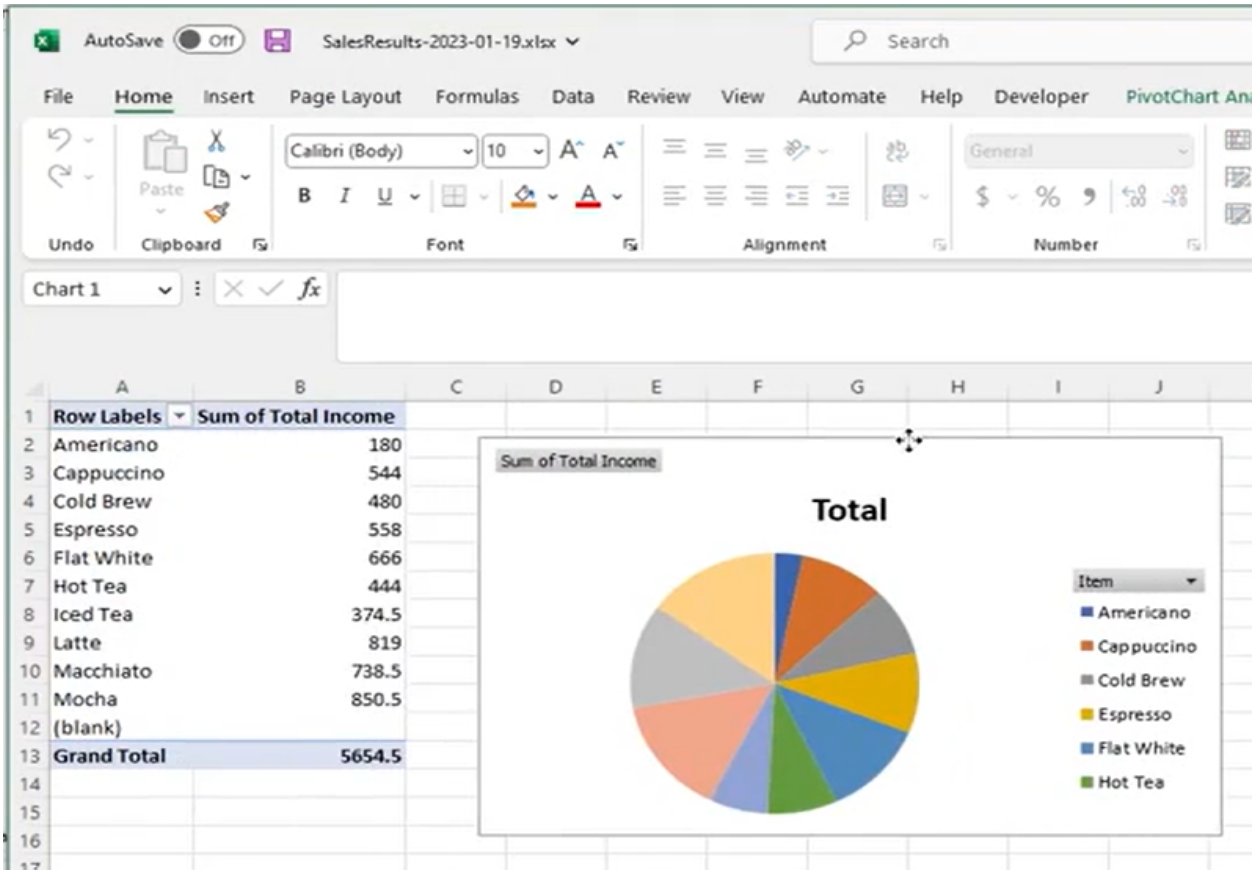


Figure 14: Pivot Table of Sales Results

The pivot table above shows a summarized view of the coffee products sold across different categories, regions, or time periods. By condensing the raw data into a clear structure, it highlights patterns such as which products sell the most, where demand is strongest, and how sales vary over time. From a marketing perspective, insights like these are essential for understanding customer preferences, segmenting markets, and identifying which products drive revenue. Marketers can use this information to tailor promotions, refine product positioning, and allocate budgets more effectively. It also helps determine whether certain products should be emphasized, redesigned, or phased out based on demand trends. When automated through RPA, this process can be repeated across multiple datasets and spreadsheets, ensuring consistent, rapid reporting that supports data-driven marketing decisions.

RPA remains widely adopted not because it is technically superior, but because it solves organizational problems that coding cannot. It bypasses IT bottlenecks by automating existing workflows exactly as employees perform them, avoiding process redesign or the need for developer resources. Crucially, RPA platforms are built to meet enterprise compliance requirements—such as SOX, SOC2, GDPR, and FDA audit standards—through tamper-proof logs, role-based access, and traceable execution, all of which would be extremely costly to replicate in custom code. Companies favor it for cost, compliance, and political reasons rather than technical elegance.

5.2.2 Digital & Neuro Marketing

These are additional subjects worth mentioning that emerge from more advanced areas of marketing. Digital marketing relies on structured platforms that automatically capture customer behaviour—impressions, clicks, conversions, bounce rates, retention, and entire customer journeys. These systems come with their own dashboards and analytics pipelines, separate from traditional CRM tools, making it straightforward to

observe how customers interact with online content in real time. Because everything is digitally recorded, the resulting data integrates seamlessly into the wider marketing and business analytics environment.

Neuromarketing, on the other hand, examines how attention, emotion, and cognitive responses shape customer decisions through methods such as eye-tracking, heatmaps, dwell-time analysis, facial-expression recognition, and interaction mapping. Its appeal lies in how these psychological reactions become measurable data signals. Once captured, they can be analyzed like any other dataset and used to refine product design, messaging, and overall customer experience. This perspective adds another dimension to understanding customer behaviour, complementing traditional operational and sales data.

In the past decade, CMOs and CFOs have increasingly relied on CAC and LTV to evaluate marketing performance and customer economics—sometimes with a level of devotion that borders on religious. Beyond this point, the discussion tends to drift away from operational reality, as these metrics are often elevated above the very conditions they are supposed to measure.

5.2.3 Marketing Strategy

My experiences across different business contexts—whether in an Australian workplace, my family's business, or through observing relatives and friends who run their own companies—have been unexpectedly valuable. They revealed a pattern: in sales and marketing, success is not strictly correlated with formal education or a traditional career path. Unlike fields that require structured progression, certifications, or technical mastery, sales and marketing often operate on a different logic. This isn't because I believe I am more capable than trained professionals—in many ways, this domain sits outside my natural strengths, as it relies heavily on emotional intelligence, persuasion, and human behavior rather than technical reasoning.

What stands out is that many of the most effective marketers and salespeople I've encountered never completed formal education. Yet the results speak for themselves. They consistently outperform academically trained professionals—not because of theory, but because of instinct, adaptability, and an intuitive understanding of people. Even when applying modern frameworks—supported by academic literature and AI-assisted research—the strategies that produce real-world results often come from individuals with no degree. Their methods are pragmatic: treat customers exceptionally well, create comfort, invest in experience, provide clean and appealing facilities, design environments for children, attract younger audiences with targeted promotions, and the customers follow.

From a digital marketing perspective, the pattern is the same. Many effective techniques emerge outside formal education—such as leveraging WhatsApp Business or Telegram for sales, using simple automated customer service systems, managing sales operations through Google Sheets, or integrating with delivery platforms like Grab or Uber Eats. The people applying these methods successfully are often self-taught rather than formally trained. The consistent lesson is clear: this field should not be evaluated solely through academic frameworks or credentials. Sales and marketing rely heavily on individual aptitude, intuition, and execution. **Marketing and sales rewards talent and execution first, education will only multiply effectiveness when applied by capable practitioners.**

5.3 Data Science & Business Intelligence

I am not an expert in this domain either, so the statements I make should be interpreted with appropriate caution. Tracing the data from earlier sections, it becomes clear that everything begins with traditional OLTP systems, which capture fast, transactional, day-to-day events in operations. In contrast, OLAP systems support large-scale analytical workloads, which is why database normalization matters early in the process—OLTP systems must maintain clean, consistent structures before their data can be meaningfully analyzed. From this foundation, organizations generate essential insights in areas such as accounting (e.g., financial stability) and marketing (e.g., product performance). Beyond these baseline interpretations, Business Intelligence (BI) tools identify deeper operational signals—delays, inefficiencies, anomalies, and quality deviations—that may not be visible on the surface. Business Intelligence (BI) occupies the middle layer between OLTP operational data and advanced predictive modeling. While OLAP databases provide the underlying structure for large-scale queries, BI tools such as Power BI, Tableau, and Looker transform these datasets into dashboards, KPIs, and analytical reports that allow managers to observe patterns in real time. This includes monitoring throughput, lead times, cycle times, scrap rates, cost variances, machine utiliza-

tion, customer trends, and other operational signals that are not immediately visible at the transactional level. In many organizations, BI acts as the primary mechanism for detecting inefficiencies—highlighting bottlenecks, quality deviations, and delays across departments. Importantly, BI does not attempt to predict future behavior; rather, it provides a descriptive and diagnostic view of the organization's current and historical performance. This makes BI a critical foundation for decision-making and the natural stepping stone toward more advanced Data Science techniques. Further extending this analysis, a variety of Data Science techniques enable organizations to move beyond basic reporting toward predictive capability. Neural networks—LSTMs being one prominent example—can learn complex temporal patterns from sensor logs, sales sequences, and machine data to generate forecasts, predict equipment failures, and detect anomalies before they escalate. This progression reflects a broader continuum in modern analytics: **operational events are transformed into data, data into insight, and insight into foresight.**

A wide range of forecasting models supports this transition. Classical statistical methods such as ARIMA, SARIMA, and exponential smoothing offer reliable and interpretable baselines for demand forecasting, inventory planning, and financial projections. More advanced machine-learning techniques—including Random Forests, XGBoost, and Support Vector Regression—capture nonlinear behaviors and are especially useful for predicting downtime, quality deviations, and customer responses. Deep learning architectures such as LSTMs, GRUs, 1D CNNs, and modern Transformer-based models extend this capability further, excelling at long-horizon forecasting using complex, multi-variable data streams typical in both manufacturing and commercial environments.

I recently remembered that I had just built a database where a single item recipe had multiple ingredient columns, which almost certainly violates even basic 1NF principles—so yes, I still have a long way to go in this domain. But as long as the data warehouse doesn't blow up, I don't care. On the positive side, most real-world business applications rarely require training large models from scratch, and deep learning architectures seldom need extensive manual tuning. This significantly reduces the complexity, time, and resources needed to implement practical forecasting and analytics solutions. I just need to import the libraries in Python.

5.3.1 Basic Use Case

Building on the example from *Section 4.3 MRP System*, I've often experienced managers request rough forecasts for material usage in time-series, typically relying only on moving averages as their reference point. With a small dataset like the one below, a simple linear regression model is an appropriate choice, since it provides a straightforward way to estimate outcomes based on past patterns. The dataset itself is fictional and is used purely to illustrate how data science techniques support prediction and forward-looking decision-making.

Order	**Complexity (1–5)**	**Material Weight (kg)**	**Labor Hours**
A	2	150	12
B	3	220	18
C	2	180	14
D	4	310	28
E	5	450	42
⋮	⋮	⋮	⋮
(Total $n = 25$ completed projects)			

Table 1: Sample Data of Completed Orders from MRP

This script below loads a dataset of completed projects from a CSV file and trains a simple linear regression model using scikit-learn. The model learns how labor hours relate to two factors—project complexity and material weight—based on historical data. The output given is an estimated labor hours of 26.7. This demonstrates how data science can generate predictions and forward-looking insights from operational business data.

```python
import pandas as pd
from sklearn.linear_model import LinearRegression

CSV_PATH = "simulated_projects.csv"
def main() -> None:  1 usage
    df = pd.read_csv(CSV_PATH)
    X = df[["Complexity", "Material_Weight_kg"]]
    y = df["Labor_Hours"]
    model = LinearRegression()
    model.fit(X, y)

    # New project prediction
    new_project = [[3, 250]]
    predicted_hours = model.predict(new_project)[0]

    print(f"Estimated labor hours: {predicted_hours:.1f}")
```

Figure 15: The Experience Of Doing Data Science As a Beginner

5.4 Artificial Intelligence

Similar to how web technologies gave business users a decisive edge in the 1990s and early 2000s—including my predecessor's enterprise before the dot-com collapse—modern industries are now entering an IoT-driven, Industry 4.0 environment where data is abundant and continuously generated. In this landscape, organizations are no longer merely building software systems; they are competing through the strategic leverage those systems can create.

> "The supreme art of war is to subdue the enemy without fighting."
> — Sun Tzu, The Art of War, Chapter 3

Today, the most significant form of leverage comes from AI systems capable of autonomous decision-making, multi-step reasoning, and generalizing from previously encountered patterns. DeepMind's AlphaGo and AlphaStar projects demonstrated this at scale, showing that reinforcement-learning agents can outperform humans in complex, high-dimensional environments such as Go and StarCraft II—domains requiring planning, adaptation, and strategic inference. More recent work at DeepMind and OpenAI has extended these capabilities into mathematical reasoning and problem-solving. However, these advances do not imply unconstrained intelligence. Current systems still face well-documented limitations in generalization, reasoning under uncertainty, out-of-distribution robustness, and real-world grounding (Marcus & Davis, 2019; Lake et al., 2017).

The economic implications are substantial. As early as 2013, Frey & Osborne identified a broad set of occupations susceptible to automation through machine learning, pattern recognition, and rule-based processing. Elements of their prediction have already begun to unfold: repetitive data preparation, clerical routines, low-skill software development tasks, and structured decision workflows are increasingly handled by AI models or automated agents. The trend is evolutionary rather than abrupt—task-level automation is expanding steadily, even where full job replacement has not occurred.

I have followed this trajectory since my undergraduate years—well before the release of ChatGPT—and years of research papers, debates, and coursework have reinforced my confidence in where the field is heading. Even if one argues that today's AI market exhibits characteristics of a financial bubble, the underlying technological leverage is undeniable. AI now automates routine reasoning, eliminates repetitive data entry, and executes operational logic at speeds and scales unattainable by human labor alone. The barrier to technical creation has dropped dramatically; individuals can now "vibe-code" complex systems with AI

assistance. Yet this accessibility does not reduce competition—it amplifies it. A more educated, more adaptive workforce means individuals who fail to leverage AI will be systematically outperformed by those who do, especially when the latter already possess strong analytical or engineering skills. In this environment, AI is not merely a productivity boost; it is a force multiplier.

5.4.1 LLM Architecture

Artificial intelligence encompasses far more than large language models. The models referenced earlier in the Data Science section—including classical machine-learning algorithms, probabilistic models, and modern deep-learning architectures—are all forms of AI capable of learning from data, adapting to errors, and improving performance over time. However, a defining characteristic of the current technological landscape is the widespread deployment of LLM-based systems such as ChatGPT, Grok, Claude, and DeepSeek.

Large language models are, at their core, statistical sequence predictors. They learn patterns from extensive training corpora that include heterogeneous sources, some of which may contain inaccuracies, contradictory information, or low-quality material. Contemporary training pipelines implement multiple layers of filtering, alignment, and post-training refinement to reduce the influence of unreliable data. Nevertheless, because **these systems generate outputs based on probabilistic inference rather than deterministic reasoning** or verified knowledge retrieval, the possibility of hallucination—producing confident but incorrect information—cannot be completely eliminated.

As of November 2025, the prevailing view among industry researchers and practitioners is that general-purpose chatbots should not be treated as authoritative substitutes for trained professionals in high-stakes domains such as medicine, law, engineering, or finance. While LLMs can provide valuable assistance, synthesis, and exploratory insight, their outputs are not consistently accurate, rigorous, or context-aware enough to be relied upon as primary sources of professional advice.

5.4.2 MCP Protocol

Model Context Protocol (MCP) is a standardized framework developed by Anthropic and was introduced in November 2024. It enables AI models to seamlessly connect with external tools and data sources without requiring custom integrations for each platform. By serving as a universal protocol, MCP ensures that AI applications can access real-time, contextually relevant data in a secure, scalable and efficient way. (Geeksforgeeks, 2025)

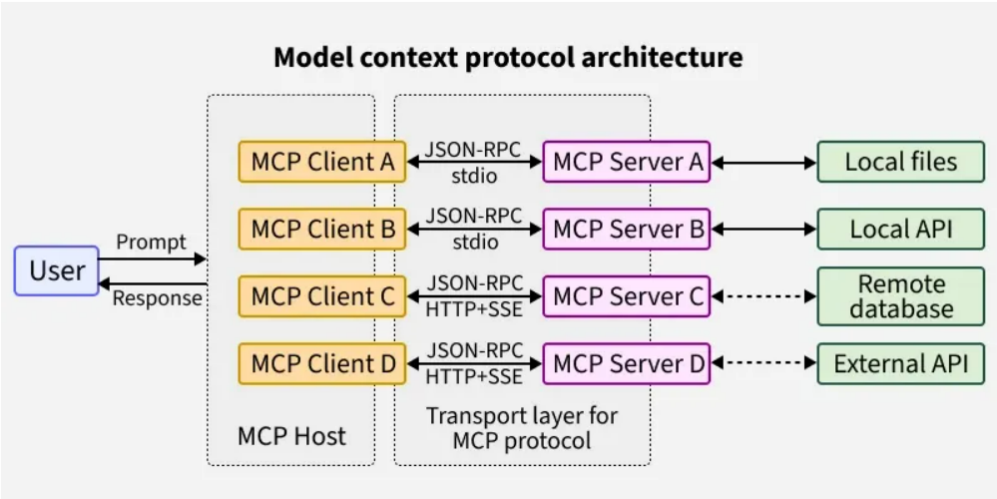


Figure 16: MCP Protocol from (https://www.geeksforgeeks.org/)

This ultimately reduces hallucination by grounding the model's responses in deterministic tool outputs rather than probabilistic prediction. **We are essentially attempting to constrain the solution space.**

5.4.3 Retrieval-Augmented Generation (RAG)

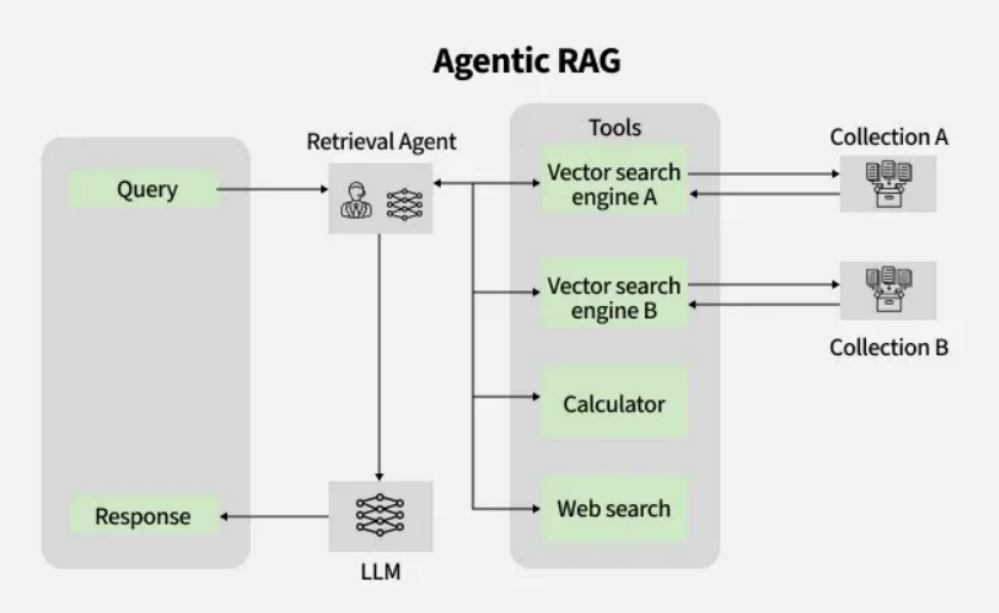


Figure 17: Single Agent RAG from (https://www.geeksforgeeks.org/)

Next, we examine the retrieval process. Retrieval-Augmented Generation (RAG) is a hybrid architecture that combines large language models with external knowledge retrieval to improve factual accuracy and reduce hallucination. Instead of relying solely on the model's internal parameters, RAG dynamically retrieves relevant documents or structured information and conditions the model's response on that retrieved context. This enables the system to generate more grounded, context-aware, and verifiable outputs, particularly in domains requiring specificity, up-to-date information, or domain expertise.

Again, the goal is to shift the system from purely probabilistic inference toward more deterministic, verified sources. By eliminating unreliable inputs—such as low-quality or contradictory blog content—the model can generate responses grounded in credible evidence. This approach also enables the system to evaluate conflicting information rather than reproduce it uncritically. **Ultimately, this reflects an effort to further constrain the problem space.**

5.4.4 Project Execution

Piecing these components together results in a standard *Agentic Retrieval-Augmented Generation (RAG)* architecture, all executed in Python. The system demonstrates how agent-based LLM workflows can surpass one-shot generation and instead produce structured, verifiable, and evidence-grounded research outputs. The results from my project are available in the APOLLO report file in the repository:
https://github.com/anthonysinggih-alt/business-systems.git.

Throughout the development process, the following key competencies and system execution was shown:

- **Architecting an Agentic AI System**
  - Designed a modular system with clear boundaries for orchestration, prompting, retrieval, and reasoning.
  - Used role-based agents instead of one large prompt, allowing separate phases for retrieval, synthesis, and critique.
  - Standardized communication using structured, machine-readable outputs (e.g., JSON).

- **Multi-Stage Reasoning Pipeline**
  - Implemented a step-by-step workflow mirroring a research process:

    $$\text{Query} \rightarrow \text{Retrieve} \rightarrow \text{Filter} \rightarrow \text{Synthesize} \rightarrow \text{Review}$$

  - Added a critique stage to catch unsupported claims and improve clarity before final output.
- **Retrieval and Knowledge Handling**
  - Retrieved information from multiple AI research sources and normalized it into a consistent format.
  - Improved retrieval quality using query expansion, deduplication, and lightweight relevance scoring.
- **Reasoning with Uncertainty**
  - Labeled information as evidence, inference, or unknown to separate facts from assumptions.
  - Used softer language when evidence was weak instead of generating confident speculation.
- **Reliability Controls**
  - Kept intermediate reasoning steps visible to support debugging, validation, and repeatability.

In summary, the resulting system demonstrates that Retrieval-Augmented, agentic LLM workflows can approximate elements of scientific reasoning by combining structured retrieval, multi-agent specialization, controlled pipeline stages, and explicit epistemic awareness. This moves beyond simple generative text production toward a framework for reproducible, machine-assisted academic research.

Research today continues to explore new AI architectures, such as asynchronous execution, which boosts agent performance by parallelizing perception and generation to increase throughput while maintaining accuracy. (Zhang et al., 2025)

5.4.5 Reflections on AI Architecture and Design

In exploring modern AI system design, I initially assumed that most technical or analytical problems could be reduced to a deterministic pipeline. My instinct was to tightly structure tasks, constrain the input space, define the format, and simply ask an LLM to fill in the content. This approach works extremely well for bounded, curated problems where the constraints are already known. In such cases, the "intelligence" lies primarily in how the human frames the task rather than in autonomous AI behavior. However, the deeper I examined concepts such as RAG, Agentic systems, and MCP, the clearer it became that these tools were not designed for the kinds of problems I was solving. They exist for domains where the solution space is effectively unbounded—such as law, medicine, scientific research, or large-scale software engineering—domains in which neither a single model nor a single human can hold or reason through all relevant information at once. These areas require iterative search, contradiction resolution, multi-step planning, validation, and role specialization. In other words, they do not demand "more memory" from a model; they demand distributed cognition, the same principle that drives human teams.

This revealed an important distinction: intelligence is not the accumulation of information but the ability to structure and refine it across multiple stages of analysis. When I created my deterministic pipeline—format specifications, selected sources, and directed LLM prompting—I was already doing the conceptual heavy lifting by narrowing the problem into a well-defined space. In such cases, a single LLM is sufficient because the task has been deliberately constrained. But problems like "find a cure for cancer" or "interpret all case law" cannot be constrained in advance; they remain open-ended and contradictory by nature. In short, the purpose of multi-agent systems is not to improve simple content generation, but to navigate problem spaces that exceed the limits of any single reasoning process. My earlier belief that all problems could fit within one mental model was wrong, it only applied to those I had already bounded through prior knowledge. The broader realization is that some domains require exploration rather than execution. Agentic AI exists for these settings: to coordinate many reasoning steps and perspectives—effectively a distributed problem-solving system—**where the problem space is infinite, with infinite solution spaces.**

6 Engineering

Before I continue, I want to credit the origin of this realization. It began during my recent visit with my family in Indonesia. As I observed and psycho-analyzed them, I started to see patterns—each person driven by their own intent, purpose, and history, all uniquely shaped by time and their circumstance. The experience was sharpened by the awareness that our time together is limited, and that some of my elders are now nearing the end of their lives. That perspective gave the insight weight.

Through intense conversations with my family and a great deal of inner work, I realized something difficult to accept: despite seven years of overseas study and significant investment, I am still at the early stages of understanding the very field I chose to pursue. Perhaps I was unfocused, Perhaps I played too much video games. But regardless of how I arrived here, I've made a decision: *I will not give up.* Recently, I found one of my childhood trophies. On the bottom, in handwriting I barely recognize as my own, I had written a goal: *"To enjoy my life."* Looking back now, through years of journaling, reflection, and moments of meditation, I understand what that sentence really meant. It wasn't about pleasure or ease—it was about autonomy. It was the early form of a Machiavellian desire: to define my own path, to choose my own direction, and to shape my life intentionally rather than passively. Maybe this pursuit of power—inner, structural, intellectual—is simply the adult expression of that first innocent ambition. And for the first time, I'm willing to admit it, face it, and commit to it fully, no matter the human cost.

In short, my current goal is to continue working in this area and obtain the remaining data needed to advance my long-term plans in business systems. It's ironic that a text written around 500 BCE already articulated strategies that still apply today—or at least match what I've only recently come to understand. As a general holds the lives of soldiers in his decisions, so does managing capitalism carry the weight of shaping human futures. War, War never changes. (Fallout, 1997).

> "Measurement owes its existence to reality; estimation of quantity to measurement; calculation to estimation of quantity; balancing of chances to calculation; and victory to balancing of chances."
> — Sun Tzu, The Art of War, Chapter 4

This section gets more technical, and readers may skip to the summary if they're satisfied with the earlier explanation in previous sections of **how business data is measured and interpreted.** Ultimately, I am trying to prove that *within the constraints of the business world*, engineering simply functions as the discipline focused on optimizing methods across the previous sections by applying science. I know specialists from my discussion in previous sections such as accounting, marketing, or data science will find my understanding shallow—not surprising, as I didn't spend a small fortune or several years of my life specializing in those fields. Regardless, I'll continue without holding back.

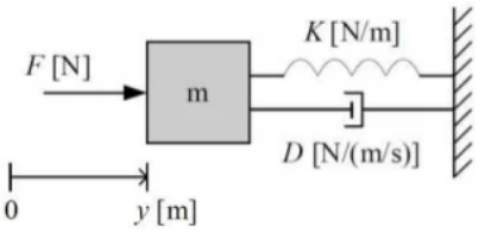
6.1 Control Theory

This subject is fundamental to most engineering disciplines and is directly related to the direction I am heading. The Laplace transform is essential because it allows differential equations that describe dynamic systems to be converted into algebraic equations, which are significantly easier to analyze and manipulate. It forms the backbone of classical control theory.

$$\mathcal{L}\{f(t)\} = F(s) = \int_0^\infty f(t)e^{-st}\,dt$$

Mechanical or electrical systems like the one shown below can be expressed using differential equations based on their physical behavior—typically in terms of acceleration, velocity, and displacement for mechanical systems, or current and voltage for electrical circuits.

- Following figure shows a mass-spring-damper-system. Where y is position, F is applied force D is damping constant and K is spring constant.



$$F(t) = m\ddot{y}(t) + D\dot{y}(t) + Ky(t)$$

- Rearranging above equation in following form

$$\ddot{y}(t) = \frac{1}{m}F(t) - \frac{D}{m}\dot{y}(t)\frac{K}{m}y(t)$$

Figure 18: Spring Damper System Differential Equations

This transformation gives us a time-invariant algebraic expression in the frequency domain, which can be manipulated like standard equations. From this form, we define the system's transfer or plant function

$$G(s) = \frac{X(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

With the plant model defined, the next step is to apply control. The system's behavior—whether stable, oscillatory, or unstable—depends on the location of its poles. The role of the controller is to adjust these dynamics to achieve the desired response. A Proportional–Integral–Derivative (PID) controller generates a control signal based on the error $e(t)$, defined as the difference between the desired reference $r(t)$ and the measured output $x(t)$:

$$e(t) = r(t) - x(t)$$

In the time domain, the PID control law is expressed as:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{de(t)}{dt}$$

Each term contributes differently to the control signal:

$$K_p e(t) \quad \text{(reacts to the current error)}$$

$$K_i \int e(t)\,dt \quad \text{(corrects accumulated past error)}$$

$$K_d \frac{de(t)}{dt} \quad \text{(responds to the rate of change of error)}$$

Applying the Laplace transform converts this time-domain expression into an algebraic form compatible with the plant transfer function:

$$C(s) = K_p + \frac{K_i}{s} + K_d s$$

This representation allows the PID controller to be combined with the system transfer function in the frequency domain for stability analysis and tuning.

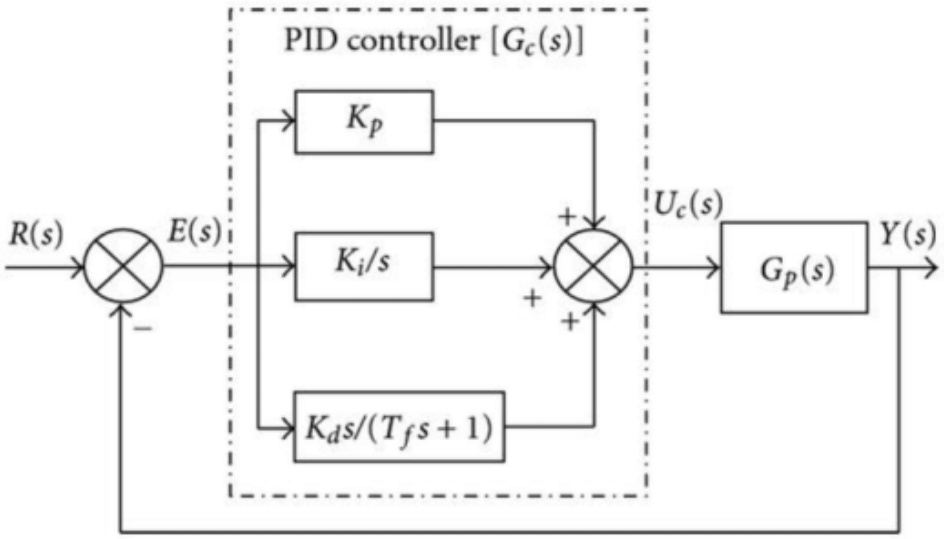


Figure 19: PID Control Designed With MATLAB

MATLAB was the primary software used throughout my engineering coursework. It was used extensively to demonstrate how variables interact and change mathematically across different subjects. In more advanced units that I undertook—such as *Advanced Control Systems* (ELEN90064) at the University of Melbourne—the curriculum applied these concepts in more complex frameworks, including state-space control and approaches grounded in Lagrangian mechanics. I barely passed and understood the concepts. In summary, this establishes the first principles that will be applied in the next section.

The reason why I am showing these equations isn't to show mathematical prowess. *Control Theory* with *Physics* helps establish the overall architectural view of why systems are designed the way they are. It also provides the foundation for understanding the first topographic layer: the physical system itself. This physical system is observed through sensors, which convert real-world conditions into signals, sometimes within embedded systems rather than industrial devices. The Programmable Logic Controller (PLC) discussed in the next section operates primarily at the control layer, processing input data and issuing commands to actuators in order to influence the physical system. In practice, many details that are rarely emphasized in university courses become critical; for example, a PLC is never wired directly to a high-power actuator, as the voltage and current mismatch would cause damage. Instead, interface devices such as contactors are used to safely separate control signals from power circuits.

1. **Physical Layer (Reality)**
   The real-world physical processes being controlled, including temperature, pressure, motion, flow, electrical current, and mechanical behavior.

2. **Sensing Layer (Measurement)**
   Devices that measure physical conditions and convert them into electrical signals, such as sensors, switches, and transmitters.

3. **Signal Layer (Estimation)**
   Processing of raw sensor signals into reliable data through filtering, scaling, smoothing, and validation.

4. **Controller / PLC (Balance)**
   The logic and decision-making layer, responsible for executing rules, managing states, handling alarms, and coordinating system behavior.

5. **Actuation Layer (Action)**
   Equipment that carries out control decisions, including contactors, motors, valves, drives, and heaters.

You can already see that this is similar to Sun Tzu's strategy in warfare I quoted above.

6.2 Programmable Logic Controller (PLC)

My knowledge is primarily theoretical, at one point I realized how little practical exposure I had when I discovered that I had never even encountered a relay while trying to make a seal logic, for example, to keep an elevator button lights on after it is pressed. Despite that, I still believe that many real-world control problems can be effectively addressed with basic PID loops and simple on–off logic. More advanced techniques such as MPC or neural-network-based control certainly exist, I'll leave them to smarter individuals to solve.

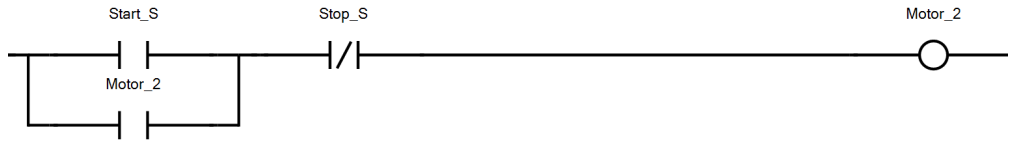


Figure 20: Latching With Relays

A variety of important PLC components, as shown in the figure above, including interlocks, sequencing logic, alarms, operating modes, and ultimately PID control, allow us to manage most industrial systems. Below is an example of an On/Off flip-flop control implemented using a Siemens PLC to regulate water levels in a storage tank.

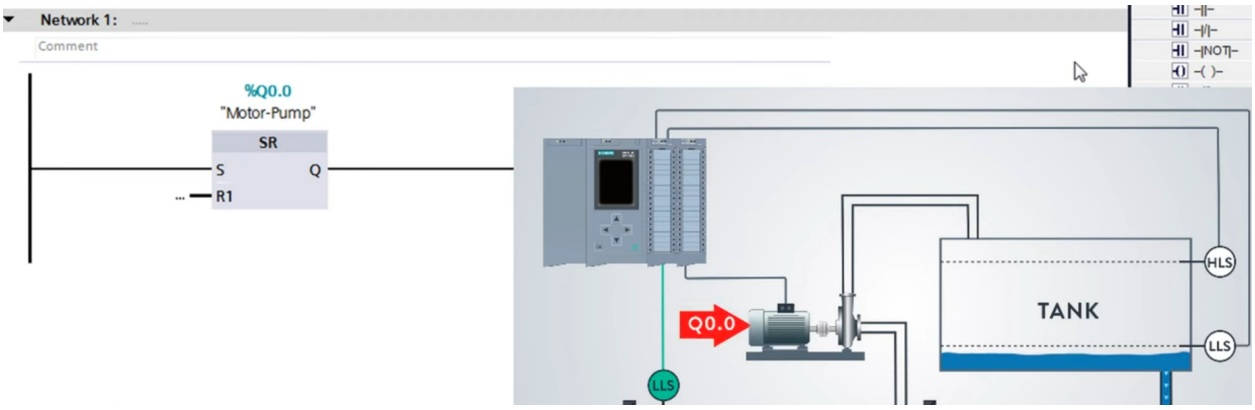


Figure 21: On/Off SR flip-flop control in Siemens PLC Software from https://upmation.com/

My insight is that control complexity does not exist because systems are inherently complicated, but because our ability to influence physical reality is indirect and imprecise. When an actuator maps cleanly and predictably to the outcome we want — for example, a pump filling a tank at a steady rate — simple On/Off control is often sufficient. As the relationship between cause and effect becomes delayed, nonlinear, or distorted by physical processes such as heat loss, inertia, or disturbance, more advanced control mechanisms like PID become necessary to continuously correct the mismatch. In other words, control theory exists not to manage complexity for its own sake, but to compensate for weak, delayed, or indirect actuation over physical systems. Better sensors and actuators outperform better algorithms almost every time. If teleportation were feasible, control would reduce to direct state assignment rather than managing acceleration or motion dynamics. **In practice, the quality of the data matters far more than the sophistication of the machine learning model**, because even the most advanced algorithm cannot correct poor measurements or unreliable inputs. It's the same realization data engineers eventually reach, and control engineers reach it too—just through a different lens. One comes from statistics and machine learning, the other from calculus and physics, but they both hit the same bottleneck: bad data and weak signals break everything, no matter

how advanced the method. At a minimum, I would like to show a PID controller implemented on a PLC below, as this represents my future end goal for proficiency alongside learning the many smaller, essential components that will allow me to begin reverse-engineering industrial machines manufactured by China.

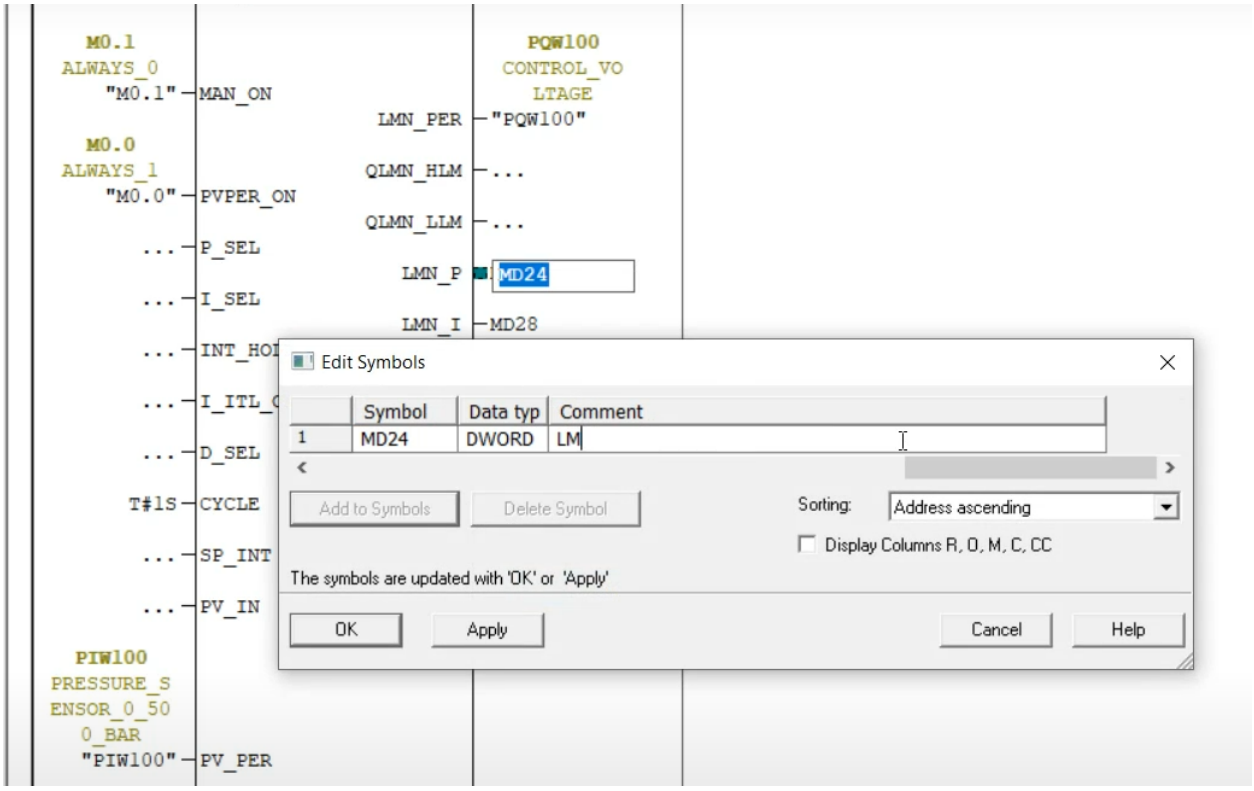


Figure 22: Pressure Control With Siemens PID Block.

As systems become more powerful, complex, or dangerous, direct control becomes increasingly rare—not for elegance, but for safety. New layers emerge to contain risk rather than to optimize performance. Across engineering, software, finance, and other critical domains, a pattern recurs: intelligence rarely touches power directly. Instead, gatekeeping layers translate intent, enforce constraints, validate actions, and prevent failures from cascading into catastrophic damage. The reason is straightforward: cognition is fallible. Humans misjudge, panic, and succumb to emotion. Artificial intelligence hallucinates and extrapolates beyond its training. Physical systems and money remain indifferent to intent, but intelligence is not. Once decision-making interfaces with execution in high-stakes environments, mediation becomes necessary—not optional. The Human–Machine Interface (HMI) is therefore not merely visual but structural. It collapses the gap between human psychology and mechanical reality, imposing constraints that wouldn't otherwise exist. This architectural principle appears at every scale: PLC controllers and interlocks sit between operators and industrial machinery; access control and audit trails separate administrators from infrastructure; MCP represents an emerging attempt to standardize boundaries between AI systems and external tools; financial regulation mediates how individuals interact with markets and capital. These layers serve a unified purpose: unmediated power breeds instability. Banks function as control systems just as PLCs regulate industrial flow. Regulation itself operates as feedback control, not bureaucracy for its own sake. The pattern holds: as capability increases, distance from direct execution tends to increase, and real authority migrates from individuals into the architecture itself. **This is safety by design.** Even if technology advanced to the point where a chip could be implanted in the brain to control machine parameters directly, the fundamental problem would remain: a human could still issue a command that causes the system to fail catastrophically. The architecture doesn't simply buffer intelligence from power—it redefines what actions are possible, what failures can propagate, and where responsibility resides. Direct control persists in some domains—surgeons

operate, pilots fly, traders execute. But in each case, the trend is clear: as stakes rise,mediation follows. The question is never whether to impose layers, but when, where, and how deeply they must go.

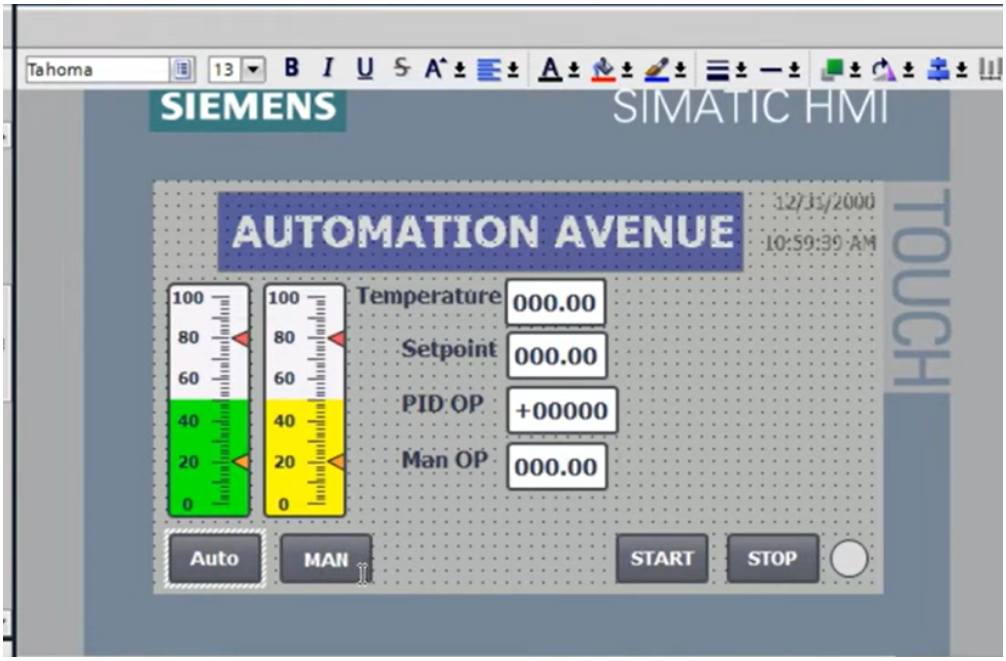


Figure 23: Siemens HMI for Temperature Control.

Please note that an HMI is not the same as a robotic controller such as a FANUC R-30iB. An HMI interfaces with a general-purpose CPU or a PLC and typically receives commands to display information and issue simple operational inputs. A robotic controller, by contrast, uses MCUs and specialized robotic operating systems to interface directly with the robot, with its controller embedded to execute far more precise, time-critical, and motion-level commands with high accuracy. Something I didn't fully realize while completing my mechatronics degree is that robotic controls operate under a fundamentally different regime from conventional industrial control systems. PLCs and HMIs are built around logic, sequencing, and state management, where precision is largely procedural and event-driven. Robotics, by contrast, operates in continuous physical space, where control is dominated by real-time dynamics, kinematics, multi-axis coordination, and high-frequency feedback loops. Instead of simply deciding what should happen next, a robot controller is constantly solving where the system is in space, how it should move, and how to correct itself in milliseconds. It is not just automation at a higher level — it is a deeper layer of control where physics, timing, and geometry become the primary constraints.

Engineering itself would likely warrant a 30-page report. Eventually, after learning a multitude of PLC logic concepts over the coming months, Eventually, I would be able to reach the Supervisory Control and Data Acquisition layer (SCADA). I am quite confident in my programming skills. Many professionals mistake job displacement of programmers by AI for a reduction in its effectiveness as a skill. I see it the opposite way: programming has become even more powerful with AI. They just don't realize it yet—we are using it to replace everyone else's jobs, starting with our own. AI has simply eliminated the use of beginner-level software thinking. Eventually, I will be able to build my own SCADA system.

6.3 Embedded Systems

While industrial systems are commonly associated with PLC-based architectures, the majority of modern machines are in fact embedded systems built around small-scale microcontrollers integrated directly into mechanical devices. These systems typically communicate using low-level protocols such as UART, SPI, and I^2C rather than industrial fieldbuses such as Modbus or PROFINET. Although modern PLC platforms

increasingly support higher-level languages such as Python, interoperability between industrial controllers and embedded devices generally requires protocol translation, gateway layers, or custom interface hardware to bridge differences in communication standards, data representations, and timing models. In practice, integration therefore involves either upgrading embedded systems to support industrial protocols or introducing middleware layers that mediate between heterogeneous devices and control architectures.

The engineering domain governing this interaction is Embedded Systems Design, which spans both hardware and software domains. Core topics include digital computer and microprocessor architectures, modeling and control of dynamic systems, models of computation, operating system principles, multitasking and real-time scheduling, resource management, hardware–software interfacing, system verification and validation, reliability engineering, functional safety, and security and privacy in constrained environments. These considerations reflect the inherently interdisciplinary nature of embedded systems, where performance, safety, and correctness must be balanced under strict computational and physical constraints. Formal treatment of these topics is provided in advanced coursework such as *Embedded Systems Design (ELEN90066)* at the University of Melbourne. If my goal is to integrate real-world devices into PLC and SCADA systems, **I should focus on communication protocols, interfacing, and safety logic** rather than designing embedded hardware from scratch unless I specifically want to work in electronics.

How? Start by choosing a simple microcontroller development board such as an Arduino, ESP32, or Raspberry Pi Pico, then install the appropriate IDE, connect the board to your computer via USB, and upload a basic test program (e.g., blinking an LED) to confirm that you can flash code onto the device. Once you can reliably program the board, learn to "talk" to the microcontroller through its pins and serial ports by reading inputs, toggling outputs, and sending test messages over UART. When working with a real device that already contains its own controller (e.g., a fan or motor drive), do not attempt to modify its firmware; instead, identify its external communication interface (such as UART, RS-485, CAN, or Ethernet) and connect your microcontroller to that port as a gateway. After that, add a Modbus library (RTU for serial or TCP for Ethernet/Wi-Fi, depending on the board), configure your microcontroller as a Modbus slave, and translate the device's native protocol or signals into mapped Modbus registers. Finally, connect the system to a PLC or Modbus master simulator, verify register reads and writes, and tune baud rate, addressing, and timing until communication is stable—at that point, you have effectively wrapped an existing device with an industrial Modbus interface rather than replacing it.

While I know this is a beginners mistake; a Raspberry Pi CPU and a microcontroller like the Raspberry Pi Pico are not just different by software or configuration but by physical design: they are built with different transistor layouts and internal logic for fundamentally different purposes. A computer CPU is engineered for high performance, multitasking, and memory throughput, relying on external RAM, operating systems, caching, and scheduling to function, while a microcontroller integrates CPU, memory, timers, and I/O on a single chip and executes firmware directly with predictable timing and low power use. You cannot meaningfully "convert" one into the other because determinism, instant boot behavior, and real-time control are properties engineered into the silicon itself, not features that can be enabled by changing software. As an example of execution, the following PLC ladder logic assigns the motor speed on command:

```
Network 1: Compute speed setpoint
|Device1_OK|----[MOVE]-----|
              IN := 1500
              OUT := Speed_Command

Network 2: Write to Modbus (conceptual)
|Always|----[MB_CLIENT]-----|
           REQ := TRUE
           MB_DATA_PTR := Speed_Command
           MB_ADDR := 40101
```

Arduino should be my logical choice because it is easy to program from a computer and supports hardware expansion through shields such as RS-485 for Modbus RTU; I can then link its registers to a Siemens PLC in TIA Portal and issue commands from ladder logic by writing to those mapped addresses.

```
int speed_setpoint;

void loop() {
    speed_setpoint = modbus.readHoldingRegister(40101);

    if (speed_setpoint > 0) {
        sendUART("SET SPEED ");
        sendUART(speed_setpoint);
    }
}
```

Without wasting years to specialize this domain, focusing on this niche concept within embedded systems **allows me to interface with low-cost, non-industrial devices**—such as encoders, fans, and pumps—produced worldwide, significantly reducing the capital required for deployment and increasing my leverage when executing large-scale business systems. This can also be accelerated through vibe coding.

6.4 Sensor Systems

Sensor systems convert physical reality into data that control, analytics, and automation depend on, but their outputs are never perfect representations of truth. Every measurement is affected by noise, electrical interference, drift, calibration error, resolution limits, and environmental factors such as temperature and vibration. A raw signal is therefore not yet "data" until it is conditioned through filtering, scaling, validation, and plausibility checks. In systems that rely on multiple sensors, redundancy and sensor fusion improve reliability not by averaging values, but by detecting inconsistency, suppressing outliers, and weighting signals based on confidence and physical constraints. Fault detection relies on monitoring whether signals behave according to expected models of the world—such as rate-of-change limits or cross-checks with related variables—rather than trusting any single measurement. One sensor provides measurement, two enable failure detection, and three or more enable fault tolerance through voting logic. Ultimately, sensor systems do not eliminate uncertainty; they manage it, ensuring that control decisions are based on estimates that are stable, bounded, and survivable under failure.

Back in the 1960s, sensors do not fail because they are inaccurate in the moment; they fail because small errors accumulate over time until the system loses contact with reality. When noisy acceleration and rotation signals are integrated into velocity and position, tiny biases and random disturbances compound, causing estimates to drift catastrophically as time progresses. Early engineers did not solve this by making sensors perfect—because perfect sensors are physically impossible—but by accepting uncertainty as a permanent feature of measurement. As systems like missiles, aircraft, and spacecraft began operating faster and farther beyond human correction, this accumulation of error became unusable, forcing the invention of statistical estimation techniques such as the Kalman filter. Rather than treating measurements as truth, these algorithms continuously predict system behavior using physical models, compare those predictions against noisy observations, and correct themselves based on uncertainty. The result is not perfect knowledge, but stable, bounded belief over time—making modern robotics, navigation, and automation possible not by eliminating error, but by preventing it from exploding.

A canonical example of the Kalman filter is shown below, which provides a mathematically optimal framework for estimating the true state of a system from noisy sensor measurements. Rather than treating sensor output as ground truth, the Kalman filter models the system as a dynamic process governed by the state equation

$$x_k = Ax_{k-1} + Bu_k + w_k$$

and the measurement model

$$z_k = Hx_k + v_k$$

where x_k represents the internal (hidden) state of the system, z_k the observed measurement, A the state transition matrix, H the observation matrix, u_k the optional control input, and w_k, v_k represent process and

measurement noise respectively. The algorithm operates in two recursive phases: a *prediction* step, in which the future state is estimated based on the model, and an *update* step, in which the estimate is corrected using new measurements. The optimal blending of model and measurement is determined by the Kalman gain

$$K_k = P_k^- H^\top (H P_k^- H^\top + R)^{-1}$$

which adaptively weights the influence of sensor data versus model prediction based on uncertainty. When measurement noise R is high relative to predicted covariance P_k^-, the filter relies more heavily on the internal model; when process uncertainty dominates, it shifts confidence toward the measurements. Through this recursive estimation process, the Kalman filter produces state estimates that are smoother, more accurate, and more fault-tolerant than raw measurements. In real systems, this enables stable control, sensor fusion, and resilience against partial sensor failure across applications such as robotics, navigation, and industrial automation.

Once we learned how to maintain reliable state estimates from noisy sensor data over time, advances in computation eventually made it practical to apply spectral analysis techniques such as the Fast Fourier Transform on continuous sensor streams from consumer devices. By the late 2010s, smartphones and wearables had sufficient processing power to combine real-time sensor fusion (often using Kalman-based estimation) with frequency-domain analysis to isolate periodic patterns associated with human motion. This same processing pipeline enables features such as pedometers in devices like the Apple Watch, where raw inertial measurements are transformed into stable estimates of movement and then analyzed to identify step rhythms and count them reliably. The Discrete Fourier Transform (DFT) is defined as

$$X_k = \sum_{n=0}^{N-1} x_n \, e^{-i \frac{2\pi}{N} kn}, \quad k = 0, 1, \ldots, N-1$$

Here is a reference implementation of a working pedometer algorithm:

https://github.com/ecruhue/Pedometer-Step-Counting-Algorithm

Larger systems such as autonomous vehicles involve far more complex mechanisms than can be meaningfully covered here, and would require many additional pages to explain in detail. However, the core point remains the same. In *Section 4.6 of Operations*, I briefly refer to *Digital Systems*, and it is important to note that these would not exist without advances in estimation, filtering, and modeling. These systems effectively simulate physical processes over time, continuously updating internal state using live data while managing significant levels of noise and uncertainty.

So why does this matter in practice? Even relatively simple filtering techniques—such as low-pass filters—can significantly improve the reliability and stability of industrial systems without requiring expensive, industrial-grade hardware and consultants. In many cases, thoughtful signal conditioning can yield meaningful performance gains at a fraction of the cost, provided it is applied with a proper understanding of the underlying system dynamics. i.e., **high-leverage knowledge for rapid success in Low-Cost Industrial IoT Integration for SMEs.**

6.5 Career Goals

My mechatronics degree didn't stay with me in the form of perfectly memorized formulas or detailed scientific procedures—much of that has faded. What remained, however, is the structure beneath it: systems thinking, interdisciplinary reasoning, and the ability to connect mechanical, electrical, software, and control principles into a unified mental model. In hindsight, that was the real value of the degree. I may have forgotten many operational specifics, but I retained the architecture of the knowledge—the "why" and "how things fit together"—rather than the step-by-step technical execution. Details can always be relearned. The underlying way of thinking takes years to develop, and that part stayed.

Shaped by both visa constraints and personal ambition, my short-term goal is to secure a position as an Automation Engineer or Technician within the next 12–18 months, preferably in manufacturing, logistics, or supply chain environments where my technical training and operational background can be applied in a practical, results-oriented way.

Over the longer term, I plan to move into a solutions architect or enterprise automation role. I am intentionally targeting work that remains technically demanding but allows me to remain office-based near my home, which is typically in the CBD. Regular travel between facilities and hands-on hardware troubleshooting, even within control systems, is not a sustainable model for me. As an alternative path, continuing within my family business remains viable if I can design or integrate systems that allow for remote work while remaining productive and capital-generating. I recently left an administrative role in a manufacturing environment to pursue more specialized office-based work, only to realize afterward that AI is rapidly compressing these roles and that immigration via administrative pathways is unworkable, even when paired with automation or AI specialization. This realization has refocused me on traditional engineering roles.

One important insight I gained over the course of my seven-year degree is that spending so much time moving across different disciplines, while maintaining only average academic performance, forced me to ask a different question: **how can I extract the maximum value from what I have learned with the minimum effort?** That question has become the central theme of my work. And to move forward, the missing components became obvious — **real-world experience and real data**.

7 Management & Leadership

With the exception of the following subsection 7.1. Anything beyond that is my own subjective opinion, and I want to be clear that I do not have formal academic training or professional experience in this area. My interest at this stage is primarily aspirational, with the intention of possibly pursuing an online MBA much later in life before I retire.

7.1 Business Planning

This is common sense: all the data is consolidated by experts in *Section 5 Analytics*, whether they are accountants, marketing, business intelligence, or data science. Meanwhile, people in operations bust their asses to actually make and sell the product with the sales team. These office people better give good insights to leadership, because that is exactly what they are paid to do and depend on to steer the ship.

Common frameworks and meeting structures referenced in industry include S&OP, MBR/QBR, OKR check-ins, KPI reviews, forecast reviews, capacity planning meetings, backlog grooming, risk reviews, budget planning sessions, steering committee meetings, stand-ups, retrospectives, pipeline reviews, go-to-market reviews, and executive alignment sessions. However, no business uses all of these—the specific mix depends on industry, company size, and operational methodology. Nearly all businesses conduct budget planning, performance reviews (monthly or quarterly), forecast reviews, and regular team coordination meetings. Sales-driven organizations prioritize pipeline reviews, while manufacturing and supply chain companies rely heavily on S&OP and capacity planning. Agile and tech companies commonly use backlog grooming, stand-ups, and retrospectives, whereas traditional industries may use more conventional project management approaches. Larger organizations and regulated industries tend to have formal risk reviews and steering committees, while smaller companies often operate with leaner governance structures. The key is that leadership selects and adapts these frameworks to fit their specific business context, ensuring that the right information flows from analytics and operations to support informed decision-making.

7.2 Philosophy

Any discussion of leadership is necessarily subjective. As I have referenced multiple times in earlier sections through the lens of military strategy, leadership manifests in many forms: some leaders are fundamentally execution-oriented, others primarily strategic, and others still are distinguished by their emotional intelligence and capacity for cohesion. Each style carries inherent strengths and liabilities, and no single model governs all contexts. Personally, my orientation aligns most closely with a Nietzschean conception of agency, in which the "will to power" is understood not merely as domination over others, but as the disciplined assertion of the self against constraint, entropy, and mediocrity. My posture is strategic, but not duplicitous; direct, to the point of fault, yet resolute in ambition. My aim is not disruption for its own sake, but the conscious effort to exceed inherited limitations and to engage the world as an adversary worthy of being overcome.

This paper emerges from a period of personal recalibration. It marks the third month since my return from Australia to be with my family, during which I have also adopted a regular practice of meditation at a local temple. The process of writing has become an instrument of orientation rather than conclusion—a way to impose structure on uncertainty and to clarify direction from within complexity. While many of my reflections remain unarticulated, I do not experience this as disorder; rather, it reflects an internal system stabilizing around its own axis.

When I ask what kind of leader I seek to become, one response is unambiguous: **my desire for autonomy exceeds my desire for affiliation or love.** This recognition alone is a form of self-awareness. Yet this impulse is not grounded in domination for its own sake, nor in chaos as an aesthetic. It more closely resembles a Machiavellian appetite for independence—the refusal to be passively shaped by circumstance. Leadership, for me, is therefore less about command and more about existential authorship: the resolve to act decisively in the face of uncertainty, accepting consequence as the cost of agency, and moving forward without the illusion of guarantee. In simpler terms: **I intend to author my own fate.**

8 Investment & Banking

Imagine I were a "quant finance" aspirant who had only been learning from YouTube for a month. Instead of drowning in abstract statistical models and complex algorithms like many self-styled Wall Street imitators—or diving into an even heavier version of data science or actuarial theory—I would take a simpler and more grounded approach. I would start by reviewing financial statements from a few local businesses using only basic accounting principles, just to understand whether a business is actually healthy. From there, I would develop a realistic sense of my own capabilities, as I attempted to do in writing this paper, and then move directly into the business environment—either as an employee or, if capital permitted, as an investor. Frankly, I would consider a McDonald's janitor more economically useful than a quant graduate who cannot secure a job; mastering advanced mathematics without producing real-world value is an exercise in vanity.

Rather than speculating on markets, I would instead begin building a diversified portfolio in the same way my grandparents supposedly did—slowly, consistently, and without any formal education—ultimately **achieving financial independence through discipline rather than theory, which I acknowledge is an area where I am personally deficient.** This is because I am aware my dad is already rich, and he is probably going to kill me if he reads this. The point is not that complex knowledge is useless, but that execution, opportunity recognition, and practicality compound faster than cleverness in isolation.

The irony is that whenever I go to the bank to buy tresury bonds, gold or mutual funds such as the S&P, people seem confused by the fact that I have little to no income in my tax returns and no formal proof of professional competence, yet still have access to these investments. Simply because I was born into the right circumstances. Perhaps this is why I find people in the financial industry unappealing to me—they feel like my natural enemies. I tend to operate through logic and blunt honesty, despite my privilege, while they are required to demonstrate competence through relationship-building, persuasion, and emotional intelligence.

9 Summary

As proposed in this paper, the appropriate mental model for understanding business systems is outlined early in *Section 3: Layers of Business Systems*. The progression of data from *Section 4: Business Operations* through its processing in *Section 5: Business Analytics* naturally leads into my discussion of future direction in *Section 6: My Engineering Knowledge*.

In summary, irrespective of the business system's complexity and mathematical reality, if you identify a skill you perform well in, apply it to create real value, and learn how to monetize it, you eventually reach the investor layer—where income is generated through ownership rather than labor. At that point, work becomes optional rather than mandatory. My own life experience in my early career reflects this inherited privilege. The path I take will be mine to shape; **defying fate, no matter the complexity.**