Project 1 – Design a dashboard

DESN 18654 Interaction Design: Behaviours

Professor: Ryan Price

Student: Yundi Guo

Feb 5, 2026

DD

Small café owner of Little Cat Café

25 years old

It is Sunday night, and DD is sitting alone in her café after closing. The past week has been busy, and she is trying to review how her business performed. Throughout the week, she has tracked sales, expenses, inventory, and orders using a mix of handwritten notes, phone memos, receipts, and spreadsheets. However, as she looks back on the week, the sense of confusion grows. Some numbers are written on paper, some are stored in her phone, and others are buried in rows of a spreadsheet. She keeps switching between records, trying to piece everything together, but she cannot form a clear overall picture.

DD wants to understand whether this week was more profitable than last week, whether today's revenue improved compared to yesterday, which days brought in the most customers, which products generated the highest profit, and which ingredients are running low. Yet when faced with long lists of numbers and scattered data, these questions feel difficult and time-consuming to answer. After a long week of running the café, DD feels mentally exhausted. The more she tries to calculate totals and compare figures, the more uncertain she becomes. She worries that she might be missing important trends, overlooking declining sales, or forgetting to restock essential items. While flipping through receipts and notes, she finds herself questioning whether she truly understands the real performance of her café.

What DD wants most is a clearer and more intuitive way to see her business at a glance — something that highlights key numbers, shows trends visually, organizes information into meaningful groups, and makes it easier to spot priorities without manually sorting through messy data.

DD's goals and motivations for using a business dashboard:

- Understand daily and weekly revenue, expenses, and net profit
- Quickly compare today vs. yesterday and identify trends
- See top-selling and most profitable products
- Feel more confident and in control of café decisions

The hardest parts of DD's business tracking process:

- Business data is scattered across multiple tools
- Difficult to see trends, patterns, and priorities
- Too many raw numbers without clear visual structure
- Fear of missing inventory or performance issues

Key insights DD looks for in a business dashboard:

- Revenue, profit, and order summaries
- Daily and weekly trend comparisons
- Top product performance insights
- Inventory status and restock alerts

Figma Prototype

https://www.figma.com/design/ZRZIAfT0pvRjlV0BtyKiop/Project-1-%E2%80%93-Design-a-dashboard?node-id=0-1&t=E2pitadWGmK9ACDH-1

https://guoyund.phoenix.sheridanc.on.ca/The-Daily-Grind-Dashboard-main/

The Daily Grind — Dashboard

Search products or inventory | Today | Theme: Pink

Showing: Today

Performance Summary

Net Profit

$1,166.75

Target: $1,150.00

↑ +9.6% vs yesterday

Revenue	Expenses	Orders	Avg Order
$1,789.50	$622.75	119	$15.04
↑ +9.5%	↑ +9.4%	↑ +9.2%	↑ +0.3%

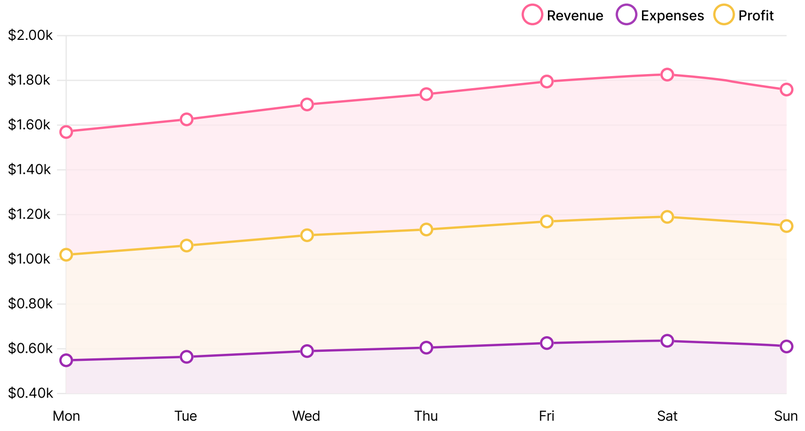
Inventory Alerts

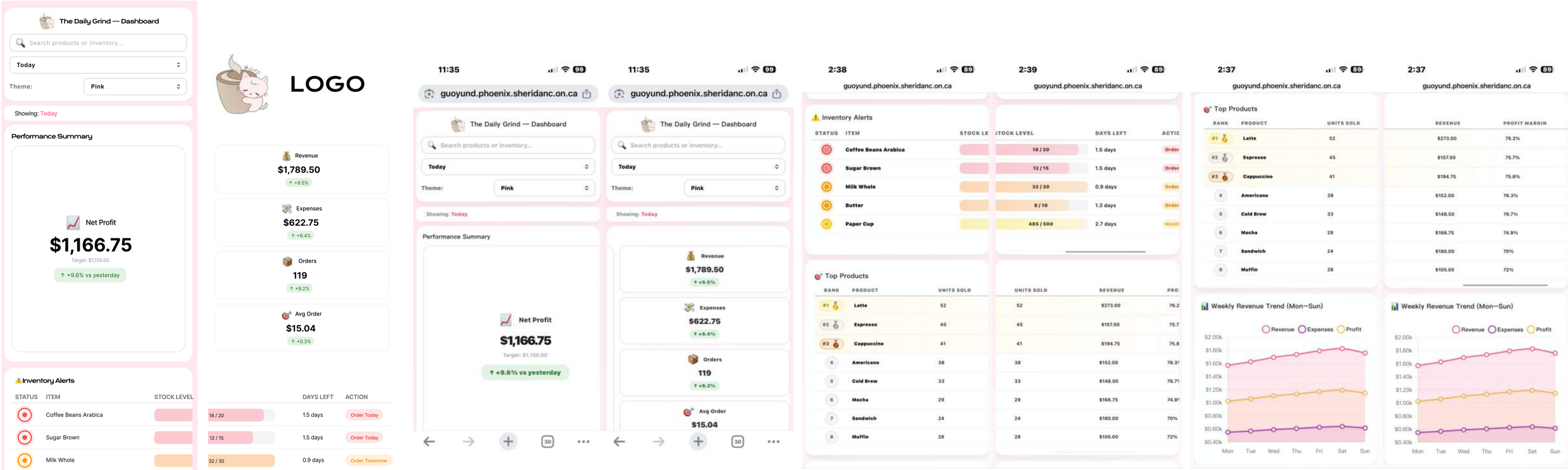
STATUS	ITEM	STOCK LEVEL	DAYS LEFT	ACTION
	Coffee Beans Arabica	18 / 20	1.5 days	Order Today
	Sugar Brown	12 / 15	1.5 days	Order Today
	Milk Whole	32 / 30	0.9 days	Order Tomorrow
	Butter	8 / 10	1.3 days	Order Tomorrow
	Paper Cup	485 / 500	2.7 days	Monitor

Top Products

RANK	PRODUCT	UNITS SOLD	REVENUE	PROFIT MARGIN
#1	Latte	52	$273.00	76.2%
#2	Espresso	45	$157.50	75.7%
#3	Cappuccino	41	$194.75	75.8%
4	Americano	38	$152.00	76.3%
5	Cold Brew	33	$148.50	76.7%
6	Mocha	29	$166.75	74.9%
7	Sandwich	24	$180.00	70%
8	Muffin	28	$105.00	72%

Weekly Revenue Trend (Mon-Sun)



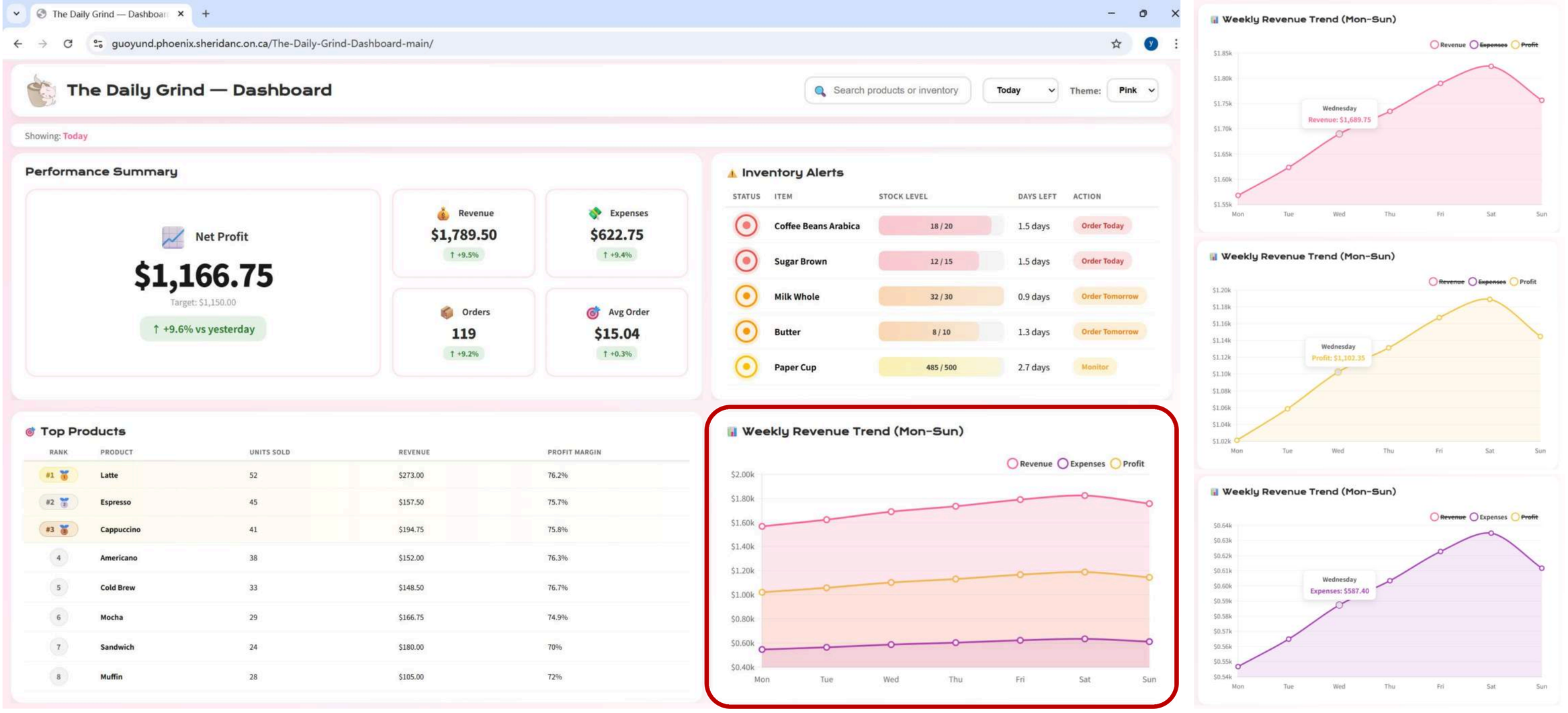


In Figma, I created the final dashboard prototype as a clear single-page layout so that users can quickly see all important information at a glance. I chose a pink color theme because I designed a cute-style logo for the café, and I wanted the overall interface to feel warm, friendly, and visually consistent with the brand.

For the visual hierarchy, I placed the most important data, such as the Performance Summary, in a highly visible area so users immediately know where to look first. I used card-style sections to organize different types of information, which helps separate content visually and makes the layout easier to scan. Rounded corners, soft colors, and spacing were used to keep the interface clean and not overwhelming.

Color is also used as a visual cue. For example, inventory alerts use icons and different colors to help users quickly notice urgent items. I tried to keep alignment and spacing consistent across the page to maintain a balanced and simple visual structure.

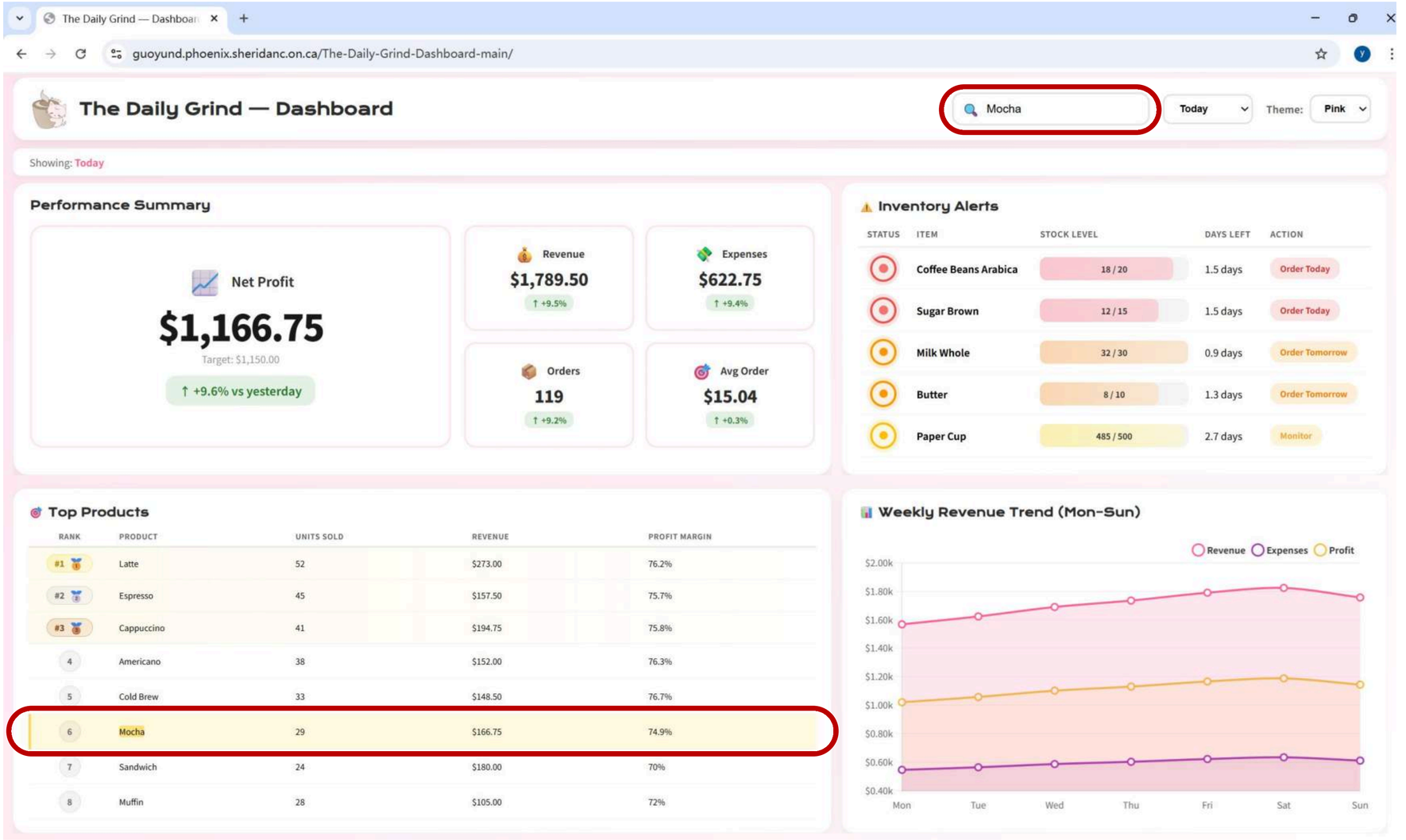
For the mobile version, I kept the same visual style but rearranged the layout to fit smaller screens. My goal was to make sure the design still feels clear, readable, and visually consistent across devices.



📊 Weekly Revenue Trend (Mon–Sun)

For the weekly revenue chart, I designed an interactive legend that allows users to turn specific data lines on or off by clicking them. When a user clicks a label (Revenue, Expenses, or Profit), the selected line becomes hidden with a strike-through effect, helping users focus only on the data they want to analyze. I chose this interaction because small business owners often want to compare specific metrics without visual overload.

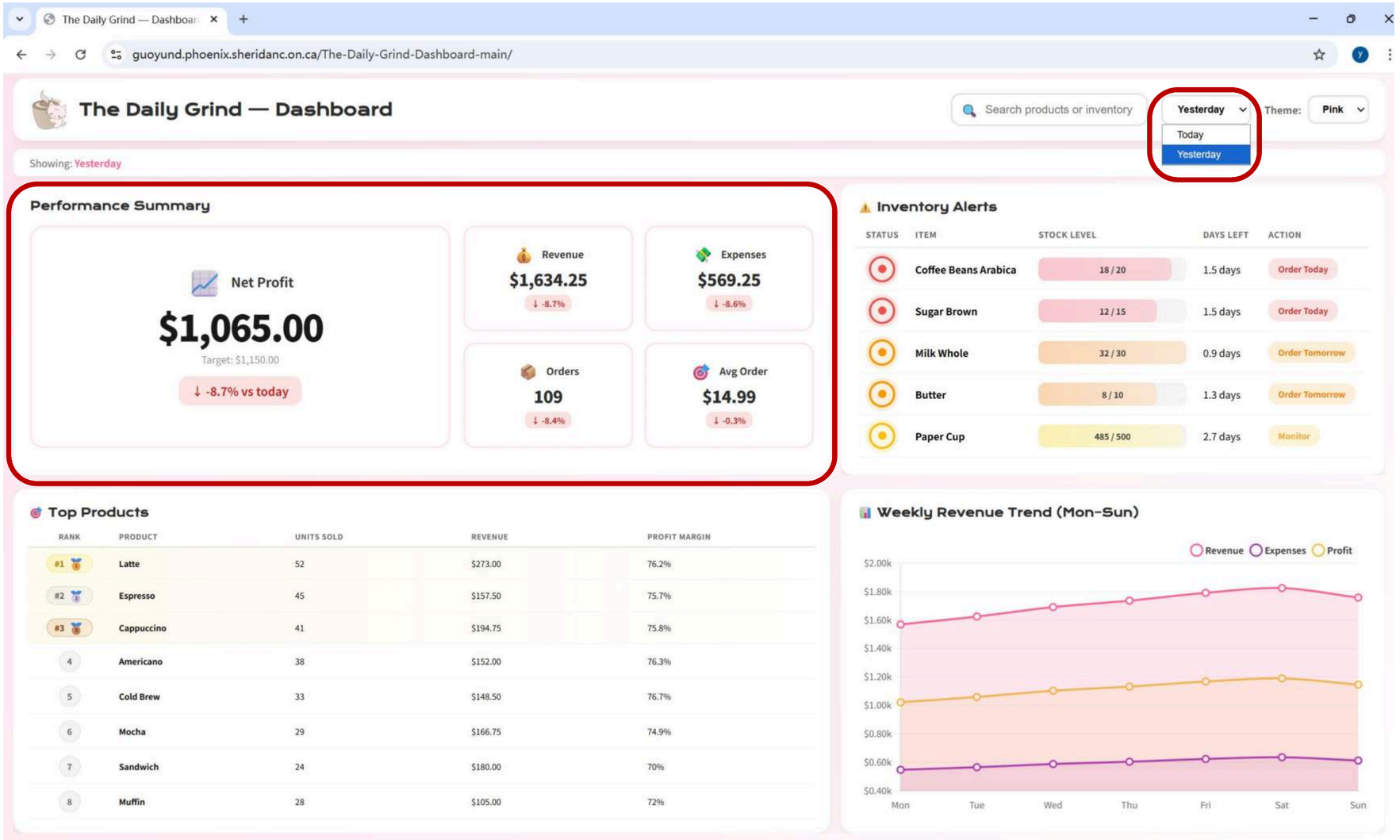
Additionally, hovering over the chart displays exact numerical values. This helps users quickly understand precise performance without needing to interpret the entire graph. I personally prefer this interaction because it makes the data feel clear, responsive, and easy to read at a glance. Profit is calculated as Revenue minus Expenses to provide a clearer financial overview.



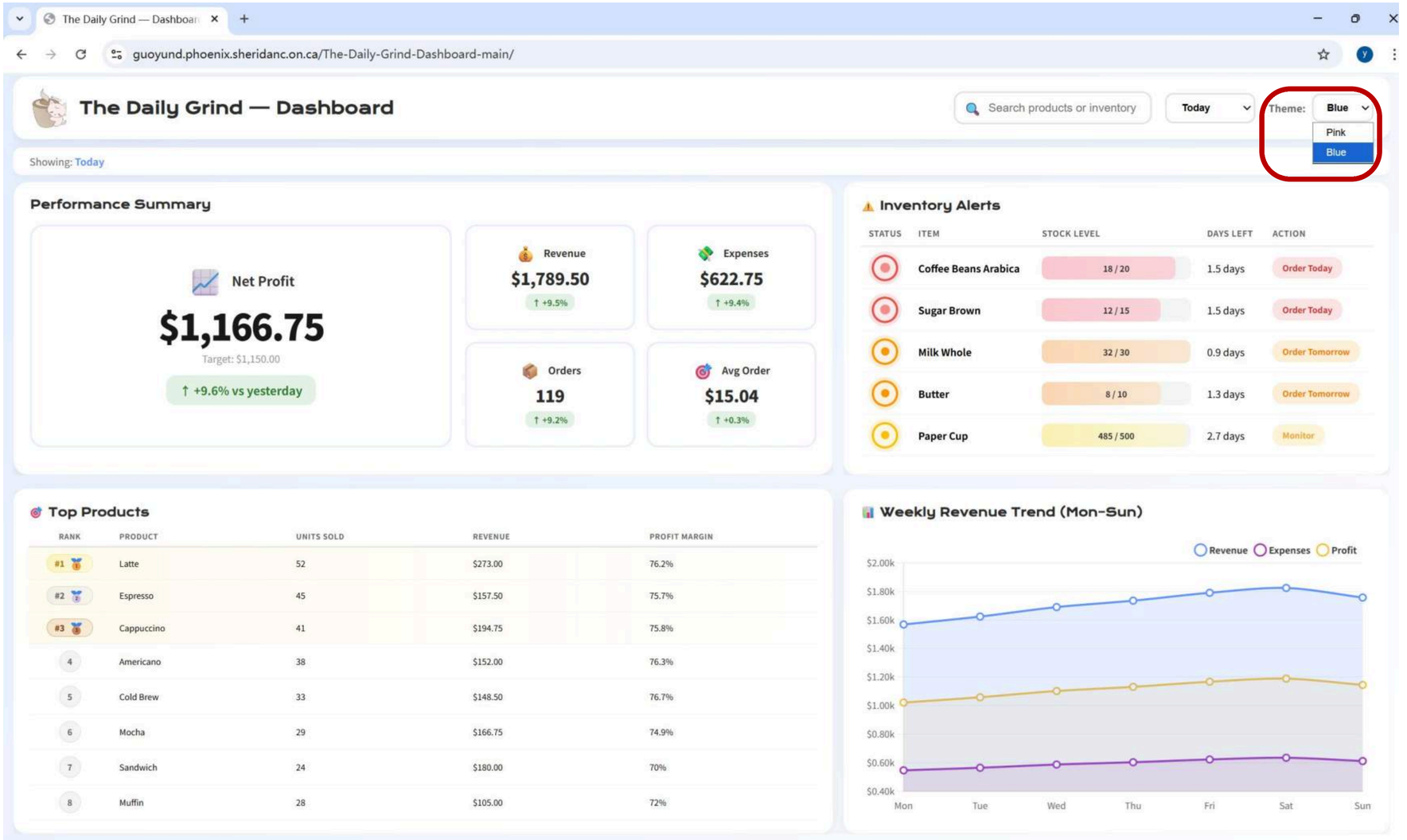
🔍 Search Function

I added a search feature to help users quickly locate specific products or inventory items. Instead of manually scanning through tables, users can simply type the name of an item and the system will highlight matching results in yellow.

The goal of this interaction is efficiency. Small café owners usually multitask and may not have time to scroll through long lists. Highlighting results visually reduces cognitive load and helps users immediately recognize where the item is located.



📅 Time Comparison
(Today / Yesterday)
Another interaction allows users to switch between today's data and yesterday's data. I included this because comparing performance between different days is an important part of decision-making for small business owners.
By simply selecting “Yesterday,” the dashboard updates automatically. This reduces the need for opening multiple pages or reports and makes comparison more intuitive and immediate. I wanted users to feel confident and informed without needing extra steps.



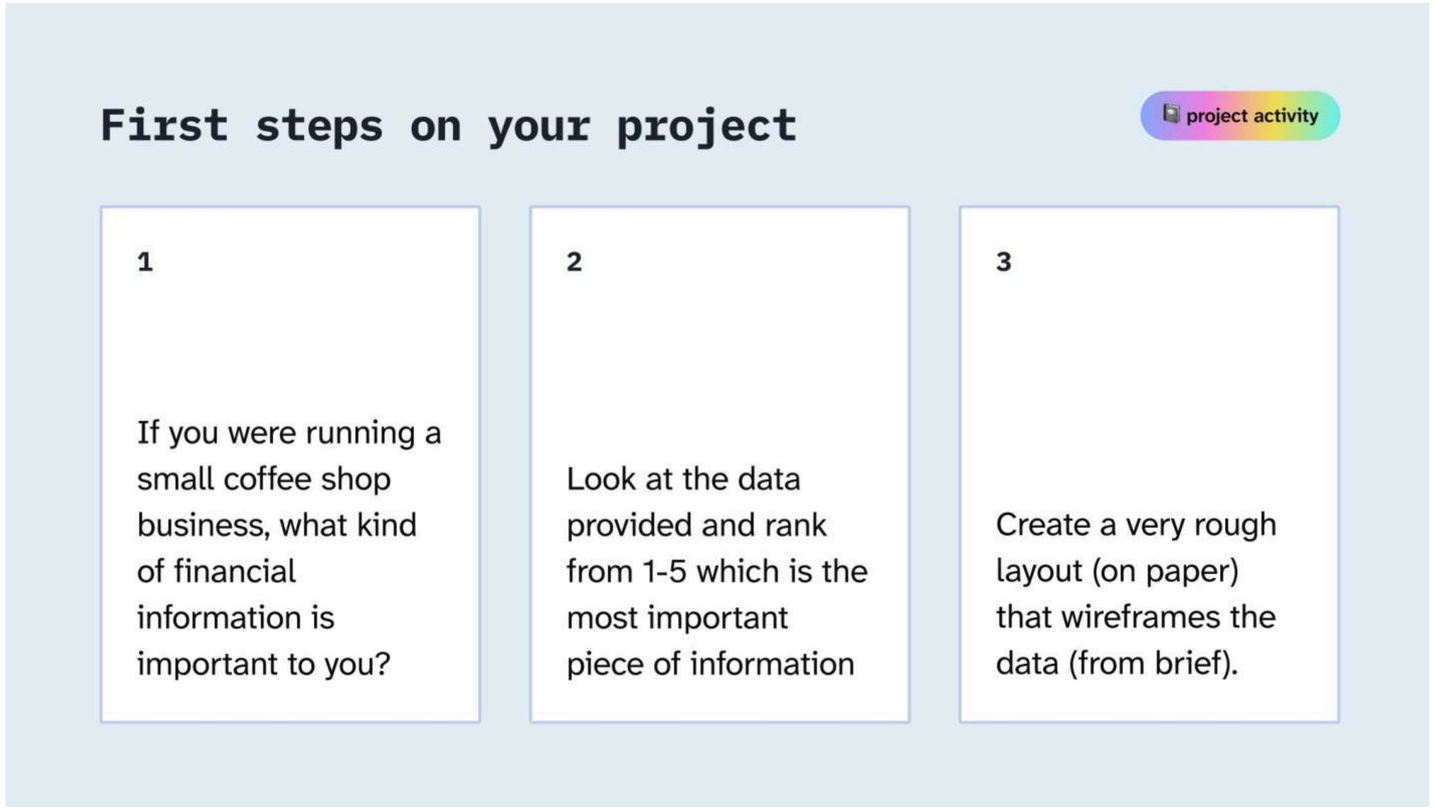
🎨 Theme Color Switching

I added a theme color option to introduce flexibility and emotional comfort into the interface. Sometimes users may feel tired looking at the same color scheme for long periods. Switching between pink and blue themes allows users to change the visual mood of the dashboard.

Pink creates a warm and friendly atmosphere, while blue provides a calmer and more focused feeling.

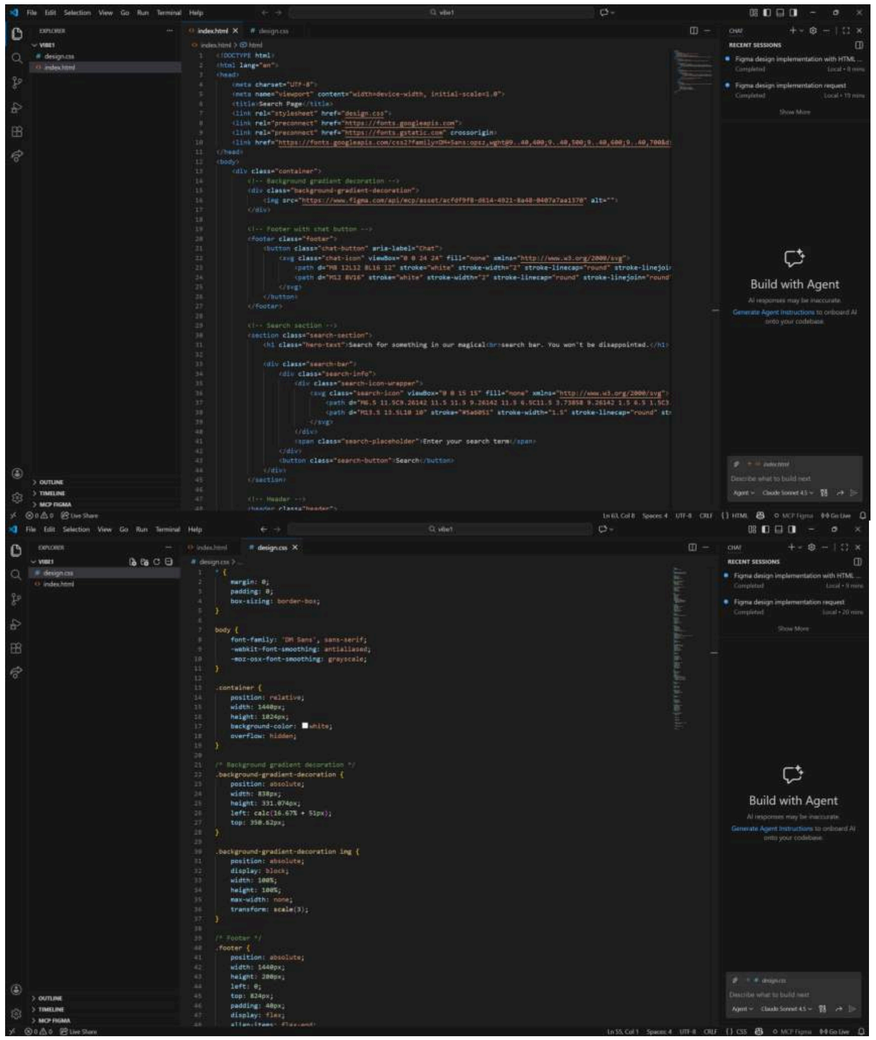
Although this feature is simple, it adds a playful and personalized experience that makes the dashboard feel more human and less rigid.

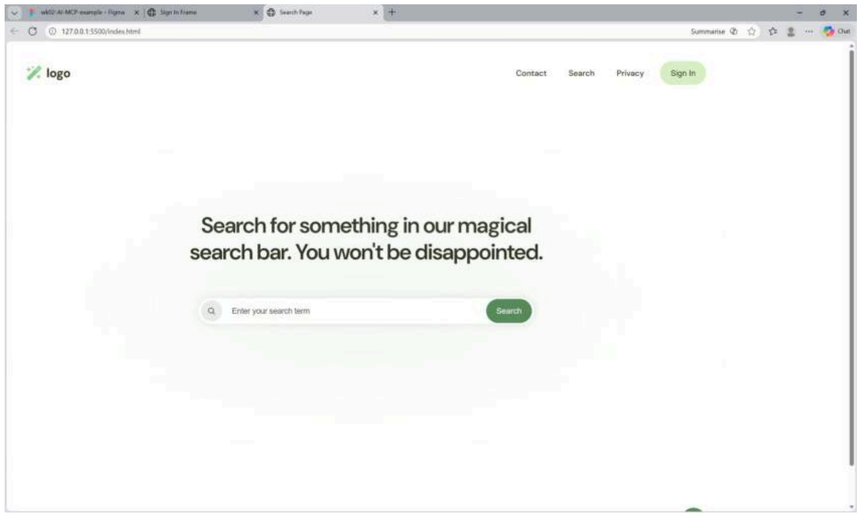
In-class activites

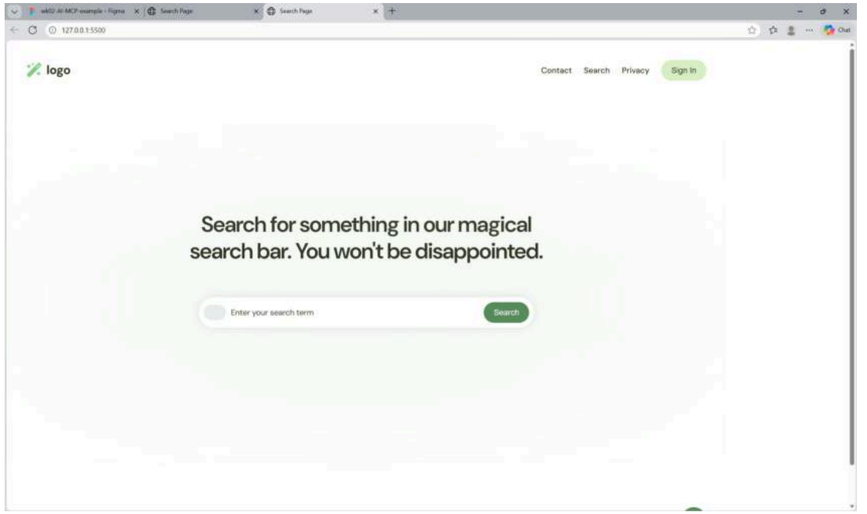


1.If I were running a small coffee shop, the most important financial information would be key performance metrics such as revenue, net profit, and number of orders. These indicators help me understand whether my business is making money and meeting daily targets. They allow me to quickly see if the shop is performing well or if improvements are needed.

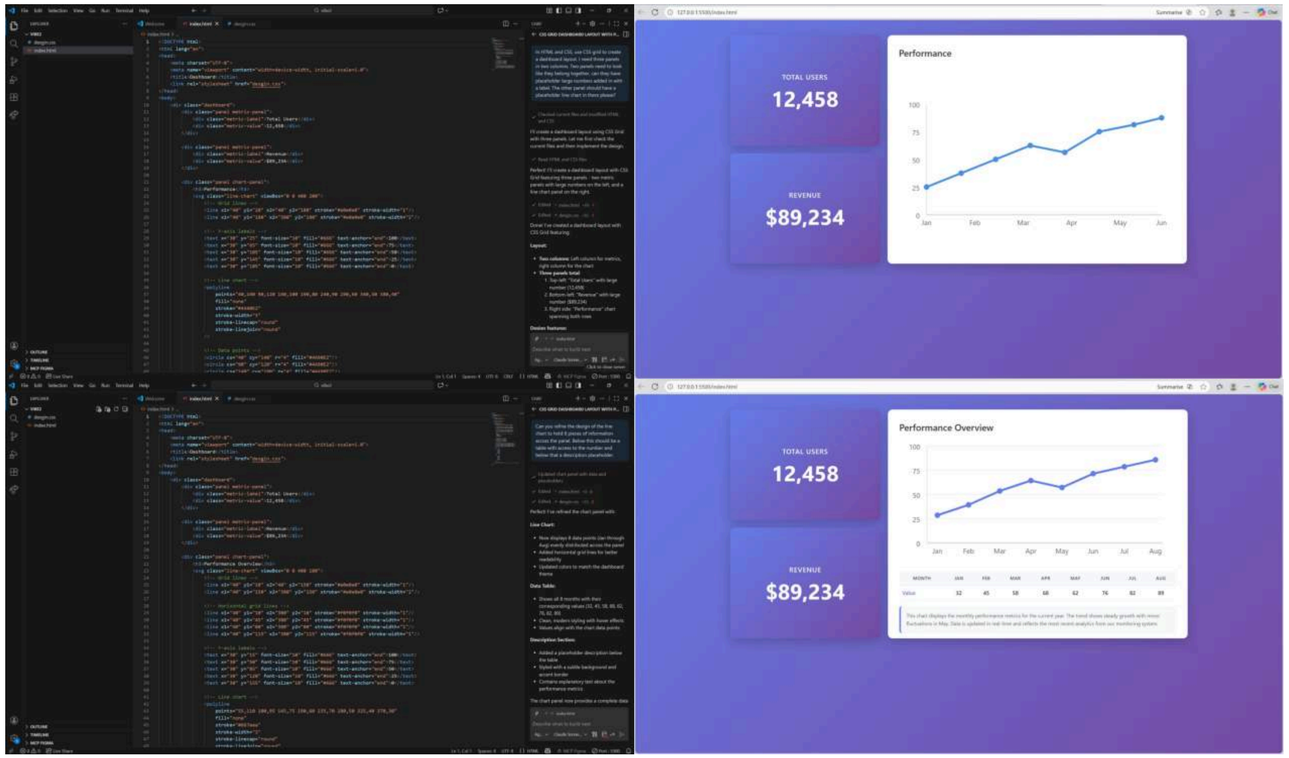
2.I ranked the Key Metrics Dashboard as the most important because it shows whether the business is meeting revenue and profit goals. Daily Performance comes second because it helps track sales trends over time. Product Performance is third because it identifies best-selling items. Inventory is fourth since it helps prevent stock shortages. Payment Method data is ranked last because it is useful but not essential for daily business decisions.







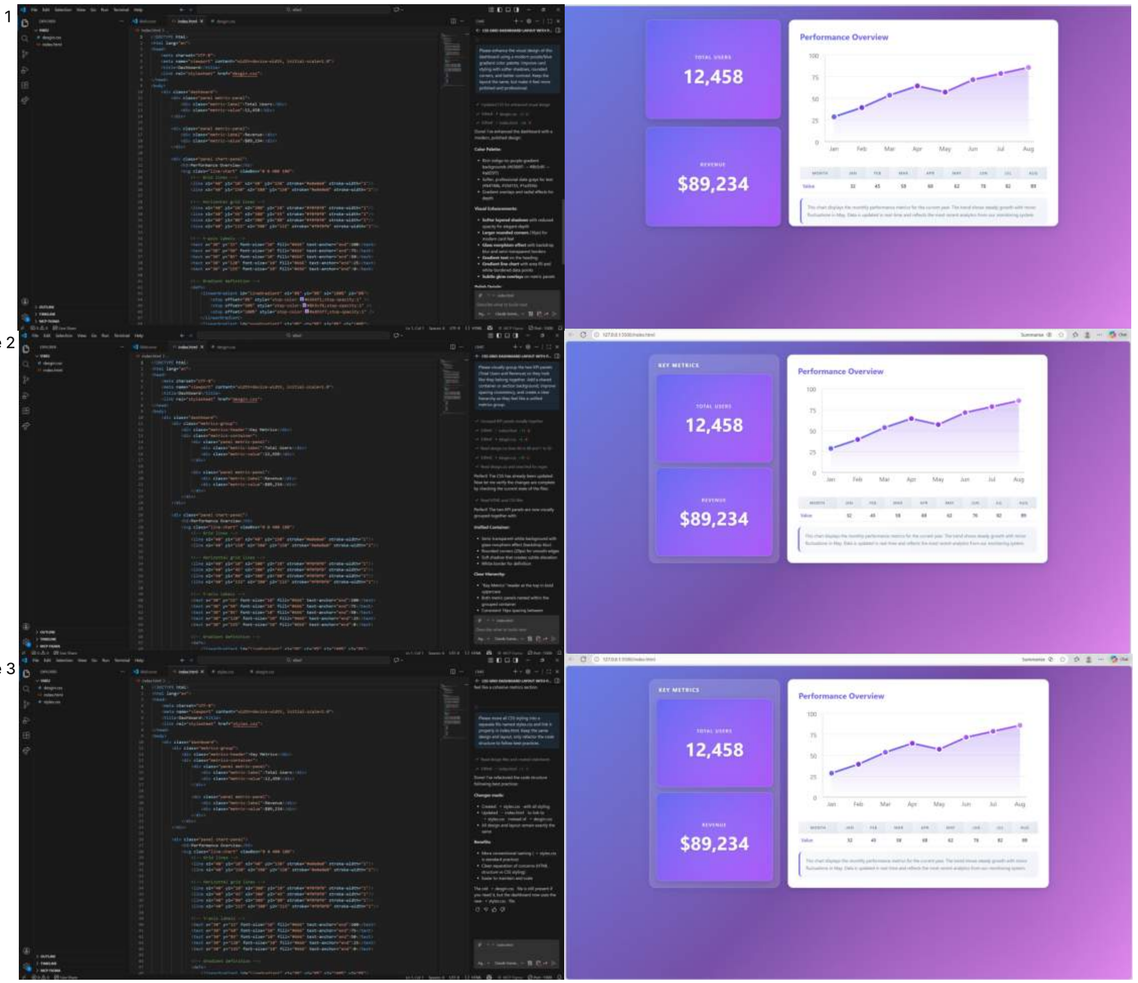
This was my first time using AI to generate a webpage from a Figma design. By following the prompts given by the instructor, I was surprised by how quickly AI could turn a visual layout into working HTML and CSS. I didn't expect AI to understand the structure of the design so well. This activity changed the way I think about web development and made me realize that AI can be a powerful tool when moving from design to code, especially in the early stages.



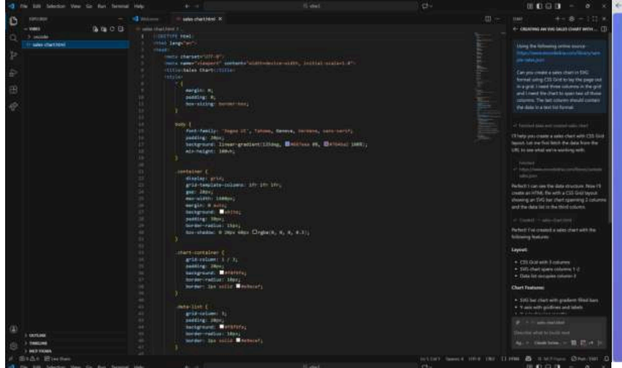
Challenge 1

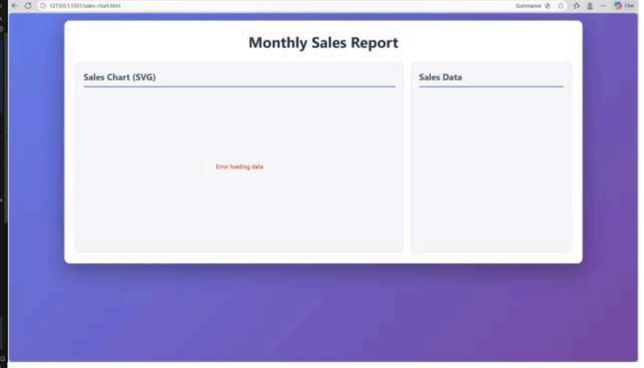
Challenge 2

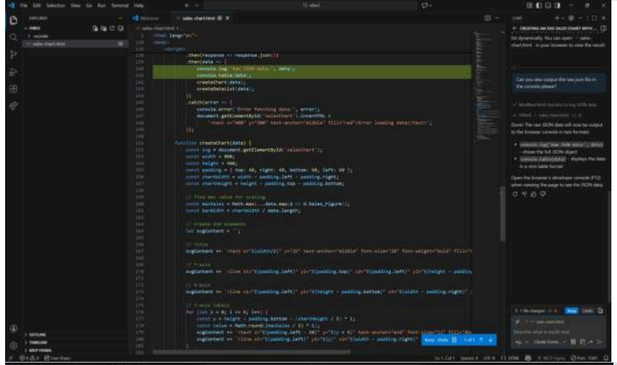
Challenge 3

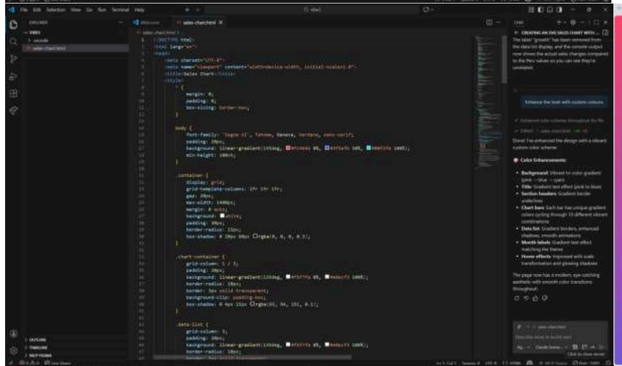


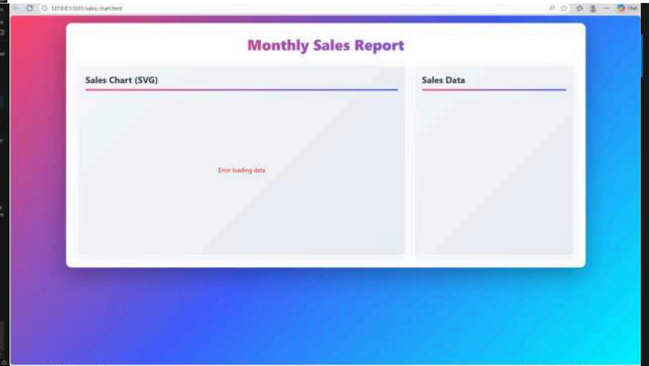
In the third activity, I used the instructor's sample prompts to generate charts and data layouts with AI. This helped me better understand how data can be visually organized instead of just shown as numbers. Exploring different prompts gave me new ideas about layout and hierarchy, and some of these ideas later influenced my own dashboard design. This activity showed me how AI can support the design thinking process rather than replace it.

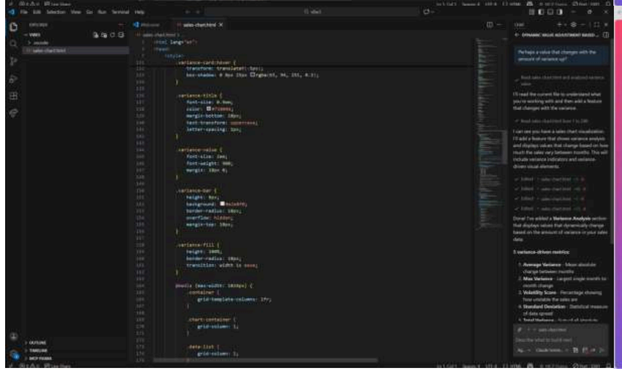


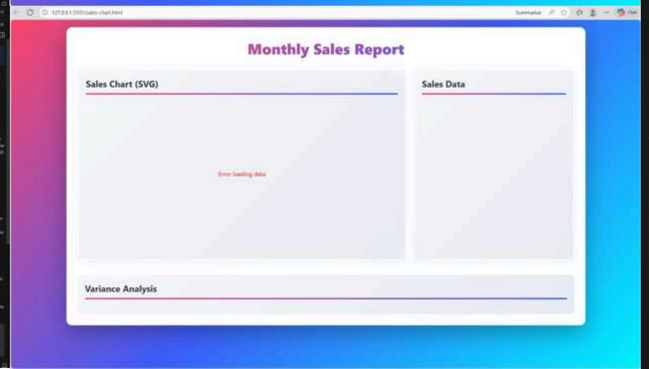


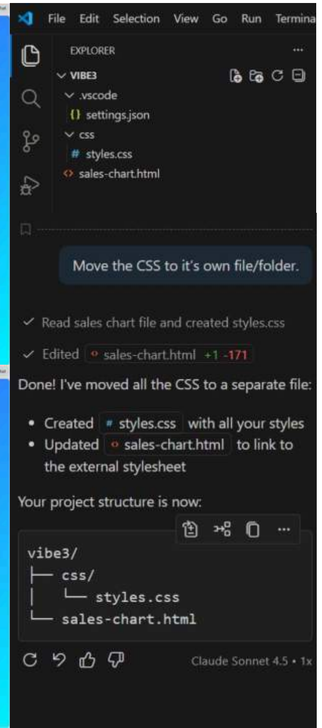












The fourth activity helped me understand how important clear prompts are when working with AI. Using the examples from class, I noticed that when instructions were more specific, the results were much closer to what I expected. In some cases, the AI even produced unexpected but interesting outcomes. This made me realize that learning how to clearly explain ideas and goals is an important skill when collaborating with AI in design work.

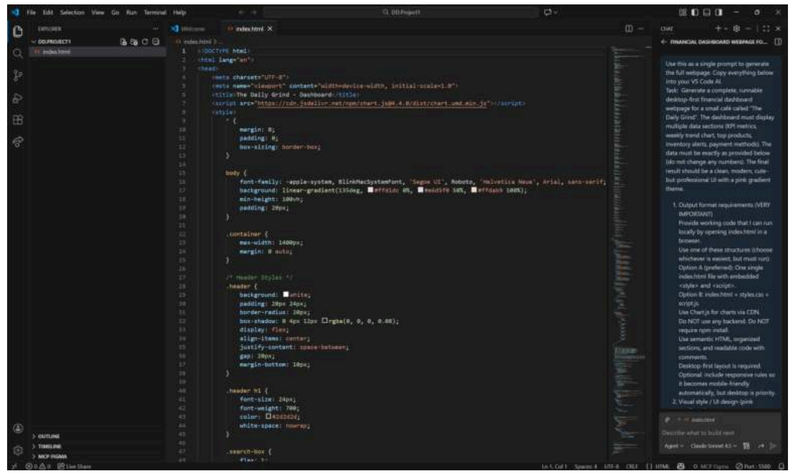
Citations

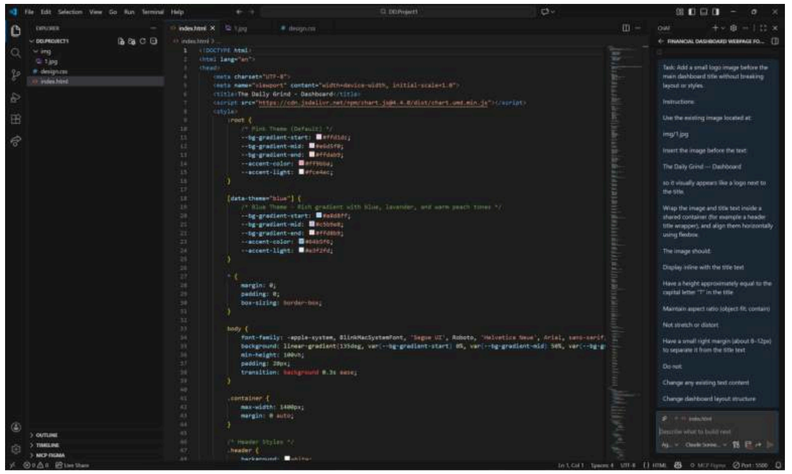
Sheridan College. DESN18654 Interaction Design: Behaviours — Week 2 Lecture Slides: “Generate Code from Figma with AI” and “Vibe Coding with AI.” Winter 2026. Course materials provided in class.
Sheridan College. DESN18654 Interaction Design: Behaviours — Week 3 Lecture Slides: “Vibe Coding with AI.” Winter 2026. Course materials provided in class.
Encodedna. Sample Sales JSON Dataset (sample-sales.json). Used as reference data for SVG chart exploration. Accessed during Winter 2026 semester.
Chart.js. Chart.js Documentation. JavaScript charting library used to implement data visualization components in the dashboard. Accessed Winter 2026.
OpenAI. ChatGPT. Generative AI tool used for writing support, editing, and structuring project documentation and reflections. Accessed Winter 2026.
Anthropic. Claude Sonnet 4.5. Large language model used through VS Code AI chat to generate and refine HTML/CSS/JavaScript dashboard code and interaction logic. Accessed Winter 2026.

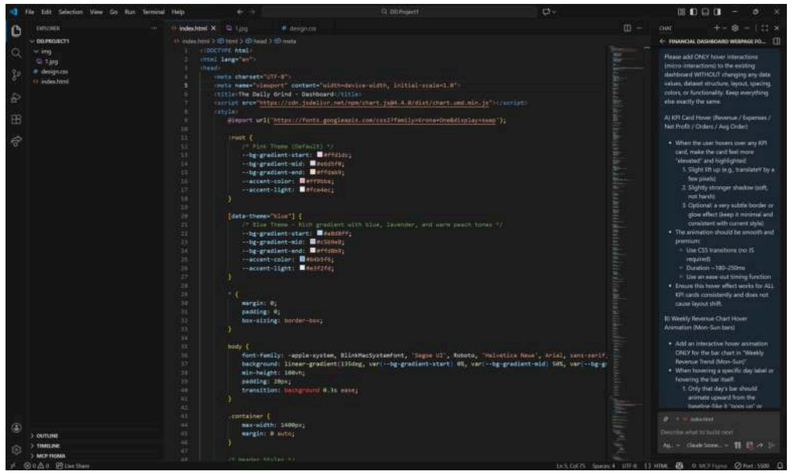
AI

AI Tool 1 — VS Code AI Chat (Claude Sonnet 4.5)
Citation
Anthropic. Claude Sonnet 4.5 (large language model). Used via VS Code AI chat. Accessed 2.5.

How I used it:
I used VS Code AI chat to generate a complete runnable HTML/CSS/JS dashboard page based on a detailed prompt and fixed data values. After the first output, I used follow-up prompts to improve interactions (search behavior, date options, theme switcher, and mobile responsiveness) while keeping the existing UI and all numbers unchanged.







Key prompts used:

- Prompt 1 (Full page generation): “Use this as a single prompt to generate the full webpage… Generate a complete, runnable desktop-first financial dashboard webpage for ‘The Daily Grind’… Use Chart.js via CDN… (data values below must not change)… Now generate the full code.”
- Prompt 2 (Interaction logic refinement): “Context: I already have a working dashboard page… Goal: Improve interaction logic without breaking anything… Search should highlight & navigate, not filter & empty… Date dropdown only Today/Yesterday… Theme switcher Pink/Blue using CSS variables + localStorage… Mobile overflow fixes…”
- Prompt 3 (Header alignment + remove search ‘x’): “GOAL: make Theme dropdown same size/style as Search + Today… remove weird ‘x’… align three controls… disable built-in cancel button for input[type=search]… Edit ONLY index.html and design.css…”

Use this as a single prompt to generate the full webpage.
Copy everything below into your VS Code AI.
Task:
Generate a complete, runnable desktop-first financial dashboard webpage for a small café called "The Daily Grind". The dashboard must display multiple data sections (KPI metrics, weekly trend chart, top products, inventory alerts, payment methods). The data must be exactly as provided below (do not change any numbers). The final result should be a clean, modern, cute-but-professional UI with a pink gradient theme.
1) Output format requirements (VERY IMPORTANT)
- Provide working code that I can run locally by opening index.html in a browser.
- Use one of these structures (choose whichever is easiest, but must run):
  - Option A (preferred): One single index.html file with embedded <style> and <script>.
  - Option B: index.html + styles.css + script.js.
- Use Chart.js for charts via CDN.
- Do NOT use any backend. Do NOT require npm install.
- Use semantic HTML, organized sections, and readable code with comments.
- Desktop-first layout is required.
  Optional: include responsive rules so it becomes mobile-friendly automatically, but desktop is priority.
2) Visual style / UI design (pink gradient theme)
Overall vibe: clean dashboard, soft shadows, rounded corners, modern typography, "gentle pink gradient" background.
Color + styling rules:
- Background: subtle pink → lavender → peach gradient (very light, not neon).
- Cards: white or near-white with soft shadow.
- Borders: very light (optional).
- Rounded corners: 16–20px.
- Spacing: generous padding (16–24px).
- Typography:
  - Use a clean sans-serif (system font stack is OK).
  - Titles bold, labels smaller.
- Use consistent icon style if needed (optional), but don't overcomplicate.
Status badge colors:
- Excellent = green-tinted badge (soft green).
- Good = blue/purple-tinted badge (soft).
- On_Track = pink/purple-tinted badge (soft).
Inventory alert dot colors:
- Critical = red dot
- Low = orange dot
- Watch = yellow dot
3) Page layout (desktop-first grid)
Create a fixed-width container (max-width ~1200px–1400px) centered on screen.
Top area (Header):
- Left: title text The Daily Grind — Dashboard
- Middle: search input (placeholder: "Search...")
- Right: date selector dropdown with default Today
- Under the header, show a small text line like: Showing: Today that updates when dropdown changes.
Section order (top → bottom):
1. KPI Cards row (5 cards)
2. Main chart section (Weekly Revenue Trend chart)
3. Two-column grid section:
   - Left: Top Products table/list
   - Right: Inventory Alerts list/table
4. Payment Methods section (chart + small table)
4) KPI Cards (must include these exact fields)
Each KPI card must show:
- Metric name
- Today value (large)
- Yesterday value (smaller)
- Change % (show + sign if positive)
- Target
- Status badge
KPI data (EXACT):
1. Revenue
   - Today: 1789.5
   - Yesterday: 1634.25
   - Change %: 9.5
   - Target: 1800
   - Status: On_Track
2. Expenses
   - Today: 622.75
   - Yesterday: 569.25
   - Change %: 9.4
   - Target: 650
   - Status: Good
3. Net Profit
   - Today: 1166.75
   - Yesterday: 1065
   - Change %: 9.6
   - Target: 1150
   - Status: Excellent
4. Orders
   - Today: 119
   - Yesterday: 109
   - Change %: 9.2
   - Target: 120
   - Status: On_Track
5. Avg Order
   - Today: 15.04
   - Yesterday: 14.99
   - Change %: 0.3
   - Target: 15
   - Status: Good
Formatting:
- Revenue/Expenses/Net Profit/Avg Order show $ sign.
- Orders is just a number (no $).
5) Main chart: Weekly Revenue Trend (Mon–Sun)
Create a chart titled: "Weekly Revenue Trend (Mon–Sun)" using a bar chart.
Weekly data (EXACT Revenue values):
- Monday: 1567.85
- Tuesday: 1623.4
- Wednesday: 1689.75
- Thursday: 1734.6
- Friday: 1789.9
- Saturday: 1823.75
- Sunday: 1756.4
Requirements:
- Use Chart.js bar chart.
- Show axis labels clearly.
- Show tooltip values with $.
- Keep chart readable and clean.
6) Top Products section (sort by Units Sold descending)
Display a table or structured list with columns:
- Product
- Units Sold
- Revenue ($)
- Profit Margin (%)
Top products data (EXACT):
- Latte — Units Sold: 52, Revenue: 273, Profit Margin: 76.2
- Espresso — Units Sold: 45, Revenue: 157.5, Profit Margin: 75.7
- Cappuccino — Units Sold: 41, Revenue: 194.75, Profit Margin: 75.8
- Americano — Units Sold: 38, Revenue: 152, Profit Margin: 76.3
- Cold Brew — Units Sold: 33, Revenue: 148.5, Profit Margin: 78.7
- Mocha — Units Sold: 29, Revenue: 166.75, Profit Margin: 74.9
- Sandwich — Units Sold: 24, Revenue: 180, Profit Margin: 70
- Muffin — Units Sold: 28, Revenue: 105, Profit Margin: 72
Sorting rule:
- Sort by Units Sold from highest to lowest.
- (So 28 Muffin should appear above 24 Sandwich.)
Interaction:
- Row hover highlight.
- Optional: clicking a row can show a small detail popover or highlight (simple is fine).
7) Inventory Alerts section (show dot + action)
Display as a list or table with columns:
- Item
- Current Stock
- Reorder Point
- Days Left
- Status (with colored dot)
- Action Needed
Inventory data (EXACT):
- Coffee_Beans_Arabica — Current Stock: 18, Reorder Point: 20, Days Left: 1.5, Status: Critical, Action Needed: Order_Today
- Sugar_Brown — Current Stock: 12, Reorder Point: 15, Days Left: 1.5, Status: Critical, Action Needed: Order_Today
- Milk_Whole — Current Stock: 32, Reorder Point: 30, Days Left: 0.9, Status: Low, Action Needed: Order_Tomorrow
- Butter — Current Stock: 8, Reorder Point: 10, Days Left: 1.3, Status: Low, Action Needed: Order_Tomorrow
- Paper_Cup — Current Stock: 485, Reorder Point: 500, Days Left: 2.7, Status: Watch, Action Needed: Monitor
Rules:
- Status dot colors: Critical red, Low orange, Watch yellow.
- Sort so the most urgent appears first: Critical → Low → Watch.
- Row hover highlight.
8) Payment Methods section (chart + small detail table)
Show title: "Payment Methods"
Use Chart.js to display a pie chart (preferred) using Percentage values.
Payment data (EXACT):
- Credit_Card — Amount: 757.6, Percentage: 42.3, Transaction Count: 48
- Cash — Amount: 438.55, Percentage: 24.5, Transaction Count: 28
- Debit_Card — Amount: 334.76, Percentage: 18.7, Transaction Count: 21
- Mobile_Pay — Amount: 216.59, Percentage: 12.1, Transaction Count: 14
- Gift_Card — Amount: 42, Percentage: 2.4, Transaction Count: 3
Requirements:
- Pie chart uses the Percentage values.
- Tooltip shows: payment type + percentage + amount + transaction count.
- Next to (or under) the chart, include a small table listing all fields.
9) Desktop-first, optional responsive (nice to have)
Do desktop layout first.
Optional: add a CSS breakpoint (e.g., at 900px) to stack columns vertically on smaller screens.
10) Interaction requirements (lightweight, but must exist)
- Date dropdown changes the "Showing: ..." label dynamically.
- Search box can filter Top Products and Inventory items by name (case-insensitive).
  - If search is empty, show all rows again.
  - Filtering is only for the lists/tables; no need to change charts.
11) Final checks before output
Before printing final code:
- Re-check that all numbers exactly match the data above.
- Ensure sorting rules are applied:
  - Top Products sorted by Units Sold desc
  - Inventory sorted by urgency (Critical first)
- Ensure the page looks visually consistent and readable.
Now generate the full code.

Context: I already have a working dashboard page (The Daily Grind — Dashboard). The current UI design, layout, colors, fonts, hover effects, and all displayed metric values are correct and must remain visually consistent.
Goal: Improve interaction logic without breaking anything.
Make minimal, safe edits. Do NOT rewrite the whole file. Do NOT remove any existing sections/cards/charts/tables. Do NOT change any numbers in the dataset. Preserve all exact data values and the current look.
0) Safety rules (must follow)
1. Do not change any existing data values (KPI numbers, tables, chart data, percentages, transactions).
2. Do not delete or rename the main data object / dataset or any render functions that currently populate the UI.
3. Make changes incrementally and keep the page fully working after each change.
4. If you need to refactor search/date/theme logic, do it by adding new functions and small modifications, not by rewriting everything.
5. After edits, the page must not show blank sections and must render all current content.
Feature A — Search should "highlight & navigate", NOT "filter & empty"
Right now search filtering causes "No results for 'X'" and empties unrelated sections. I do NOT want that.
What I want search to do:
1. Nothing disappears.
   - Do not remove rows from tables.
   - Do not replace content with "No results for ...".
   - Do not clear tables or sections.
2. Instead, search should:
   - Highlight matching content (keep the current soft yellow highlight style — I like it).
   - Scroll / jump to the first match on the page smoothly (optional but preferred).
   - If there are multiple matches, keep them all highlighted.
3. If there are no matches, show a small, non-intrusive message (like a toast or tiny helper text near the search input) saying:
   - "No matches found."
     But still keep all content visible.
What counts as "match"
Search should match text across:
- Top Products table: product name, units sold, revenue, profit margin (text match)
- Inventory Alerts table: item name, status/action text, numbers
- Payment Methods: legend labels + payment table (method name, amount, %, transactions)
- Weekly chart: day labels (Monday–Sunday). If user searches "Monday", highlight the Monday bar/label instead of filtering the page.
- KPI card titles: Revenue, Expenses, Net Profit, Orders, Avg Order (if the user types these words)
Highlight behavior (very important)
- When a match occurs in a table row: highlight the matching text AND subtly emphasize the row (e.g., light yellow background or outline).
- When a match occurs in the chart: highlight the matching day bar (and/or label). Do not hide other bars.
- When a match occurs in payment methods: highlight the matching legend label and the matching payment method row in the table. Do not hide other methods.
Search UX details
- Keep placeholder text: "Search products, inventory, payments, or status..." (good)
- Add a magnifying glass icon inside the search input on the left (common search pattern).
- Keep the clear "×" button on the right (good).
- Search should be case-insensitive and match partial words (e.g., "Esp" matches "Espresso").
- Do NOT interpret random words like "Monday" as a reason to remove non-Monday content. Only highlight.
Feature B — Date dropdown should not include unavailable ranges
Currently the dropdown includes "Last 7 Days" and "Last 30 Days" but we do not have real datasets for those, which confuses users.
Requirements
1. The date dropdown should only include:
   - Today
   - Yesterday
2. Remove "Last 7 Days" and "Last 30 Days" entirely (not just showing N/A).
3. Switching Today/Yesterday should update only what your current implementation supports, but must remain consistent and not break layout.
4. Keep the "Showing: Today / Showing: Yesterday" label accurate.
(If you already built more range logic earlier, keep the code if you want, but hide those options from the UI. The UI must only show Today/Yesterday.)
Feature C — Theme switcher (keep pink default, add blue option)
I LOVE the current pink gradient background. Keep it as the default theme.
What I want
Add a small "Theme" control in the header area (top left or near the top bar, whichever fits the current design):
- Theme options:
  a. Pink (Default) — the current one
  b. Blue — a similar soft gradient vibe but blue-toned
- When user switches theme:
  - Background gradient changes accordingly
  - Optional: accents (badges/highlights) can slightly adapt, but do not destroy the current style
- Remember the selection using localStorage so it stays after refresh.
Implementation preference
- Use CSS variables + a data-theme="pink|blue" attribute on body (or root container).
- Keep all existing spacing, card styling, shadows, rounded corners.
Feature D — Mobile responsiveness fixes (important)
When I resize to mobile width, some content overflows:
- The pie chart gets too large and spills outside its card
- Tables overflow outside the card or cause layout break
Requirements (mobile)
1. Charts must stay inside their cards:
   - Any canvas/SVG should scale to container width (max-width: 100%).
   - Maintain aspect ratio and avoid cropping.
2. Tables should not overflow the card:
   - If needed, allow horizontal scrolling inside the card for tables only (not for the whole page).
   - Keep header row visible.
3. Ensure cards stack nicely in one column on small screens.
4. Keep the overall design consistent with current desktop styling (just responsive).
Acceptance checklist (must pass)
After changes:
- All original numbers and labels are unchanged and still visible.
- Search:
  - highlights matches
  - never empties sections
  - never shows "No results for ..." inside tables/cards
- Date dropdown only shows Today/Yesterday.
- Theme switcher works + remembers choice.
- Mobile view: no chart/table overflow outside cards.
Before finalizing, briefly summarize what files you edited and where you changed logic.

You are working in an existing HTML/CSS dashboard project. The UI is already correct and I do NOT want any redesign or layout changes beyond the 3 specific fixes below.

GOAL (only these 3 fixes):
1) Make the "Theme" dropdown (Pink/Blue) the SAME visual size/style as the Search input and the "Today" dropdown. Right now the theme select looks tiny and inconsistent.
2) Remove the weird small "x" icon that appears near/under the Search field (looks like a clear/cancel button). I do NOT want that to show.
3) Fix alignment: Theme select, Search input, and Today select must be perfectly aligned on one row (same height, same baseline, vertically centered). Also the magnifying-glass icon inside the Search input must be vertically centered and not floating too high/low.

IMPORTANT RULES:
- Do NOT change any data, charts, content text, or the rest of the page layout.
- Do NOT change the overall design theme colors, cards, spacing of the whole page, etc.
- Only touch the minimal HTML/CSS needed for these 3 fixes.
- If you add classes, keep them specific and do not break other elements.
- Ensure it works on desktop and mobile.

WHAT TO CHECK FIRST:
- Identify the header controls container that holds: Theme select, Search input, and Today select (whatever wrapper div currently exists).
- Confirm the Search input type. If it is type="search", browsers inject a default clear/cancel "x" button. That is likely the weird "x".

IMPLEMENTATION REQUIREMENTS:
A) Make the three controls match in height and style:
- Create or reuse a shared CSS rule for header controls (both <select> and the search <input>):
  - same height (e.g., 44px or 46px)
  - same border radius (match the existing rounded style)
  - same border color
  - same font size
  - consistent horizontal padding
- Apply this to:
  - the Theme <select>
  - the Search <input>
  - the Today <select>

B) Remove the weird "x":
Option 1 (preferred, minimal): keep type="search" but remove the browser cancel button using CSS:
- Add CSS that disables the built-in cancel/decoration for WebKit browsers:
  input[type="search"]::-webkit-search-cancel-button { -webkit-appearance: none; appearance: none; }
  input[type="search"]::-webkit-search-decoration { -webkit-appearance: none; appearance: none; }
- Also ensure no extra element is being inserted by our own HTML (no extra "x" button). If there is an extra element, remove it ONLY if it is responsible for the unwanted "x".

Option 2 (if needed): change the input to type="text" (only if the cancel button cannot be removed cleanly). Keep the same placeholder and functionality.

C) Center the magnifying-glass icon inside the input:
- Ensure the search wrapper is position: relative.
- Ensure the icon is position: absolute; left: [some px]; top: 50%; transform: translateY(-50%);
- Ensure the input has enough left padding to avoid overlapping the icon (e.g., padding-left: 44px or similar).
- The icon must look centered vertically inside the input across browsers.

D) Align the three controls in one row:
- The controls wrapper should use display:flex; align-items:center; gap:[consistent];
- Prevent the search field from collapsing on smaller screens:
  - allow the search to flex:1; but keep selects fixed width/min-width
- Make sure all three controls have the same height so alignment is perfect.

DELIVERABLE:
- Edit ONLY the existing files (index.html and design.css).
- Provide the exact code changes:
  1) show the exact HTML line(s) you changed (only if necessary)
  2) show the exact CSS you added/modified (with selectors)
- Do not include unrelated refactors.

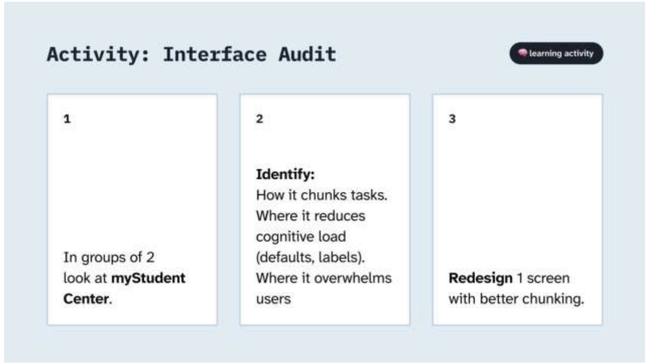
After changes, the Theme dropdown should look the same size and style as the Search and Today controls, the unwanted "x" should be gone, and all three controls + the magnifier icon should be neatly aligned.

AI Tool 2 — ChatGPT
Citation
OpenAI. ChatGPT (generative AI). Used for writing and organizing project documentation. Accessed 2.5.

How I used it:
I used ChatGPT to help me write clear and concise slide text for my in-class activities and reflections. It also helped me organize my citation/AI documentation so it matches the assignment requirements and avoids repeated wording.

What I asked it to do:
- “Write simple English slide text describing my three in-class activities without repetition.”
- “Help me format the Citations page and the AI annotated citation page based on the brief.”

Appendix



Identify: Chunking & Cognitive Load

How it chunks tasks:
The MyStudent Centre groups similar features into clear categories such as Profile, Academic Records, Classes, and Financial Services. This creates a hierarchical structure where users first choose a main category, then access more detailed options. This step-by-step structure helps users focus on one decision at a time.

Where it reduces cognitive load:
The interface uses icons, visual cards, and clear labels to help users quickly recognize different functions. Key features are visible on the main dashboard, reducing the need to remember navigation paths. The layout is organized and scannable, making it easier to locate important information.

Where it overwhelms users:
Some sections, such as the Service Requests page, appear empty or lack guidance. When users encounter blank screens without instructions, they may feel confused about what to do next. This increases cognitive load and reduces user confidence.

Within the Service Requests section, tasks are grouped into a single page where users can view or create requests.
This creates a basic task structure, but the page does not clearly break the process into smaller steps.
Redesign idea:
To improve the page, the interface could break the process into clearer steps and provide better guidance.
Suggested improvements:
Add a clear empty-state message
For example:
"You have no service requests yet."
"Click below to create a new request."
Break the task into simple steps
Step 1: Choose request type
Step 2: Fill in request details
Step 3: Submit and track progress
Add helpful visuals and explanations

- Icons to show request categories
- Short descriptions explaining what each request type means
- A visible "Create New Request" button

These changes would reduce cognitive load, make the process feel more organized, and help users understand what to do more quickly.

Proximity
Related charts and metrics are spread too far apart
Text, labels, and graphs are not visually grouped
Users cannot easily tell what belongs together

Similarity
Different chart types and data blocks use inconsistent styles
Similar data sections do not look visually consistent
Users cannot quickly recognize patterns

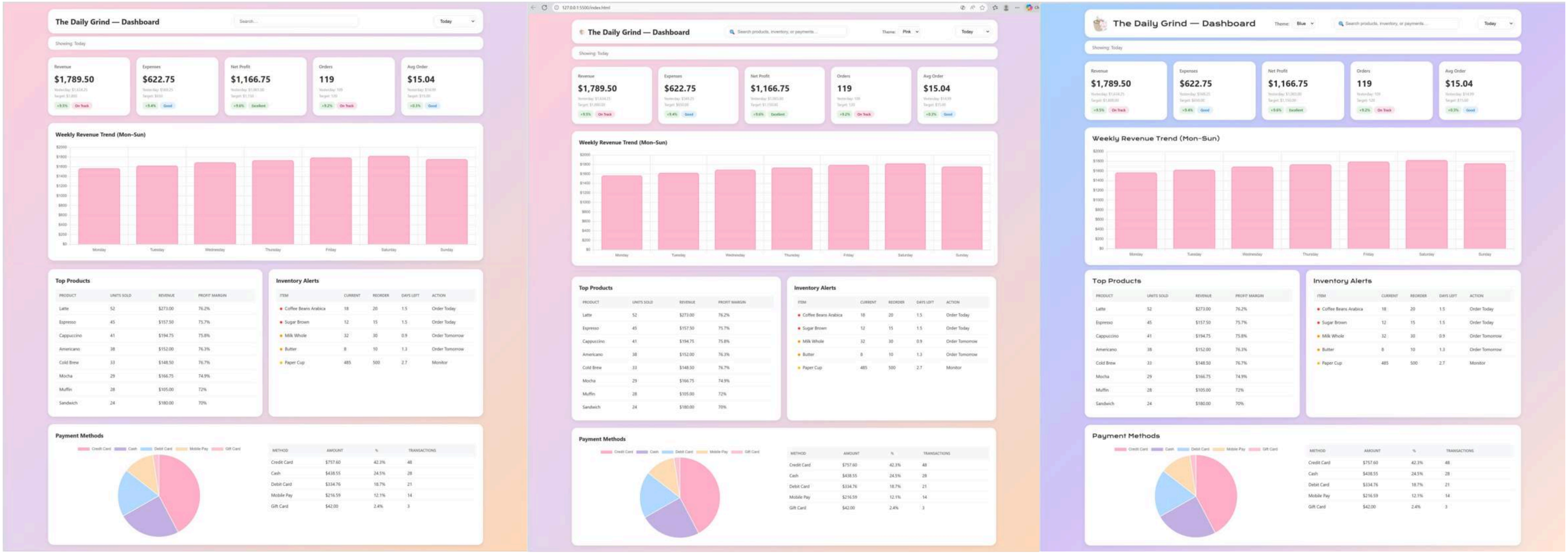
Continuity
The reading flow is interrupted by scattered layouts
The eye does not follow a clear visual path
Users must jump around to understand the dashboard

Figure–Ground
Important insights do not stand out visually
Too many visual elements compete for attention
The main takeaway is not clearly highlighted

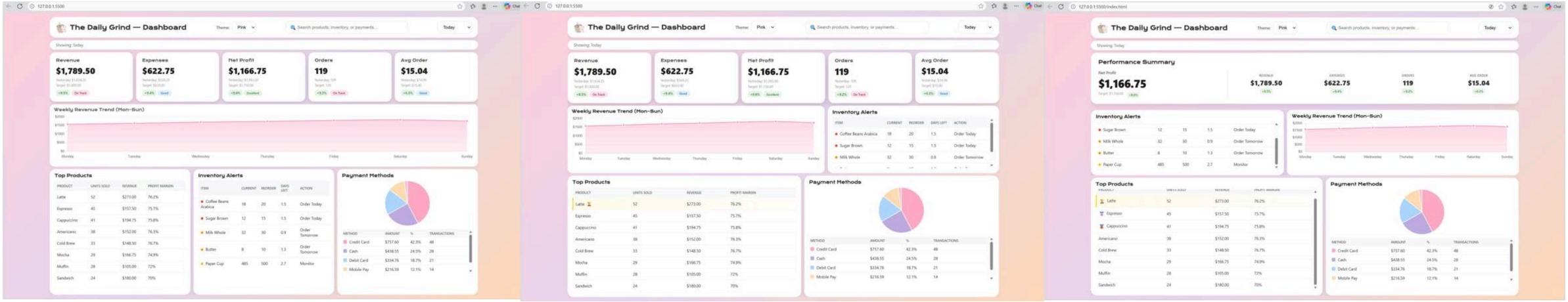
Cognitive Overload
Too many charts shown at once
Too much dense numerical data
Hard to quickly find key insights

I chose the Retention and Graduation chart section because it contains the most important insights on the page.
The current layout breaks several Gestalt principles. Related charts are placed too far apart, which weakens proximity. Similar data blocks do not share consistent visual styles, which breaks similarity. The reading flow is unclear, which disrupts continuity, and key insights do not stand out clearly from the background, weakening figure–ground.
In my redesign, I group related charts closer together, use consistent colors and styles for similar data, and arrange elements in a clear top-to-bottom reading path. I also highlight the most important metrics to make them easier to notice. These changes improve clarity, reduce cognitive load, and help users understand the data more quickly.

This stage shows the learning activity and initial analysis process.
I explored interface audit concepts such as chunking, cognitive load, and Gestalt principles to understand how users perceive structure and information hierarchy.
Through this activity, I learned how grouping, proximity, similarity, and visual hierarchy influence usability. Instead of focusing only on visual style, I started thinking about how layout decisions affect user understanding and navigation.
This phase helped me shift my mindset from decoration to behaviour-driven design.



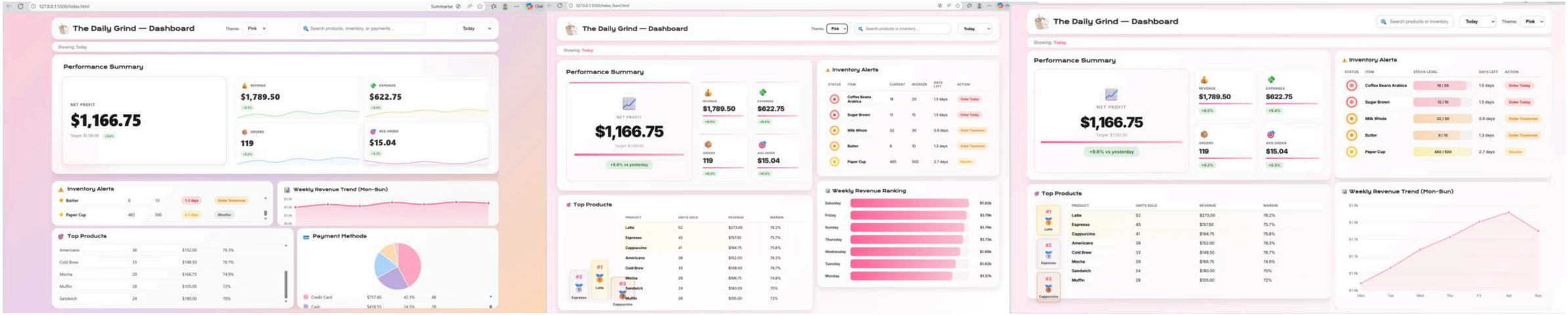
In my first design version, I tried to include all dashboard content together with vibrant colours and a strong visual style.
My main goal at this stage was to organize everything neatly and create a visually attractive interface. I focused on alignment, structure, and ensuring all data sections were present.
However, the layout became dense because too much information was displayed at once. This helped me realize that visual richness alone does not guarantee clarity.



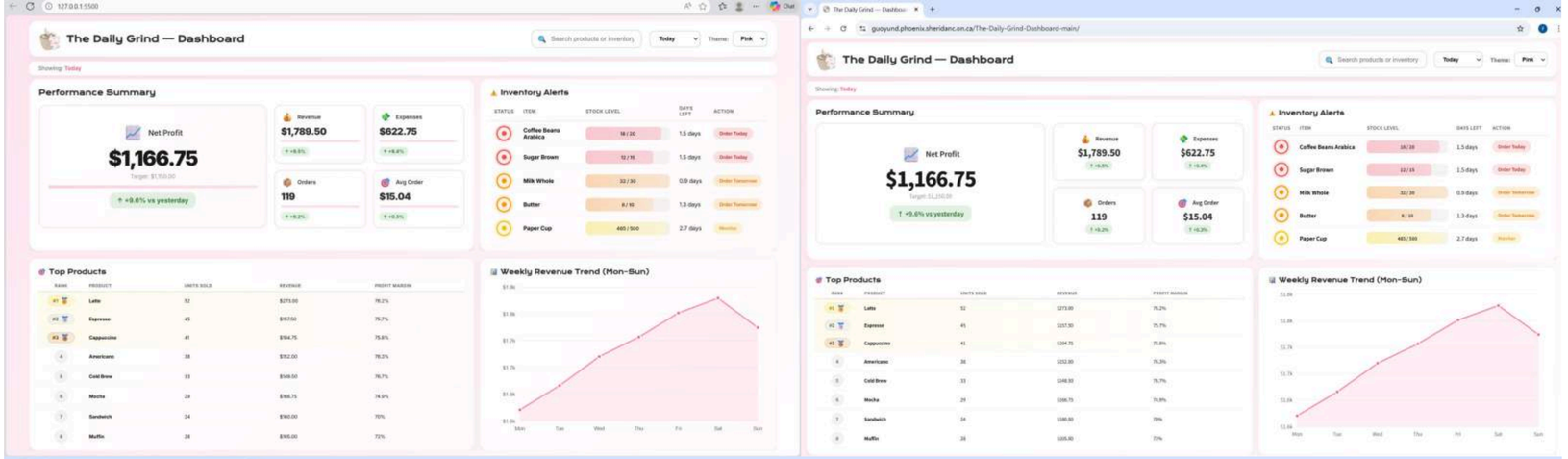
After receiving feedback from my instructor, I redesigned the dashboard into a single-page layout to reduce scrolling and improve usability.

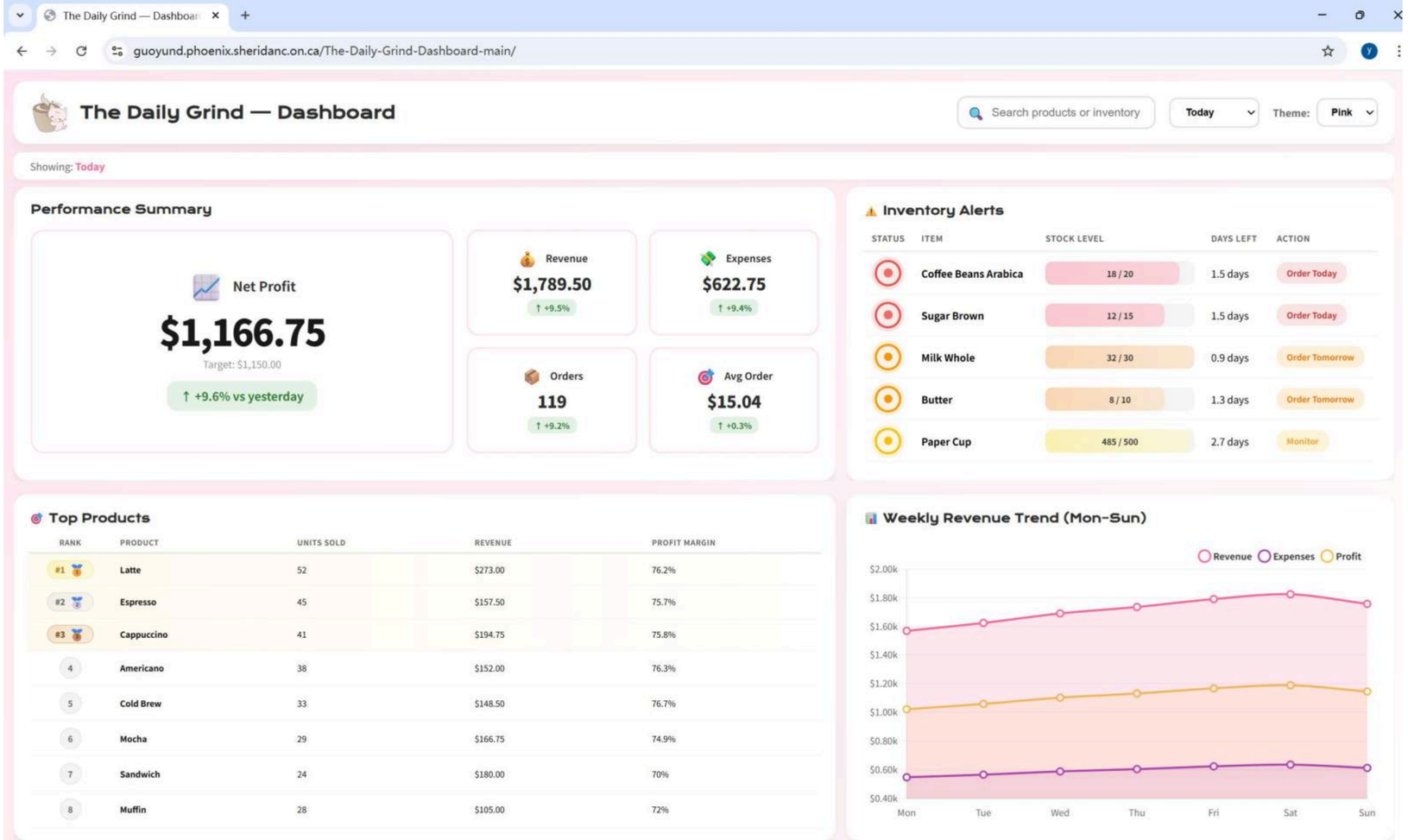
I rearranged the content hierarchy and experimented with different layouts to balance clarity and aesthetics. I also added small icons to improve visual recognition and support faster scanning.

During this phase, I explored multiple solutions to fit all information into one screen while keeping it readable. I experimented with layout adjustments and interactive ideas to manage limited space.



In this stage, I focused on improving visual hierarchy and information emphasis.
I explored different chart arrangements, icon placement, and spacing strategies to highlight key insights while maintaining clarity. The goal was to make important data immediately noticeable without overwhelming the user.
This phase involved more detailed design thinking, where small visual adjustments were used to improve usability and readability.





The final version reflects a series of refinements based on previous experimentation and feedback.

I adjusted layout spacing, chart presentation, and visual balance to create a cleaner and more structured dashboard. Small visual changes helped improve consistency and reduce cognitive load.

This stage focused on polishing the overall experience and ensuring the interface communicates information clearly and efficiently.

1

Plan:
Choose one learning goal for today.

My goal for this project was to learn how to design a clear and effective dashboard. I wanted to practice organizing key data in a way that is easy to understand at a glance. I also aimed to improve my skills in Figma, especially creating charts and layouts, and to explore how AI tools and prompts could support my design process. Overall, I wanted to better understand how to choose the most important metrics and present information in a way that feels clean, readable, and trustworthy.

2

Monitor:
Mid-task check-in, am I on track?

While working on this project, I actively applied what I learned in class and spent a lot of time experimenting with different layouts and chart combinations. At times, I felt unsure about which information was most important and how much content should be shown on one screen. Through feedback from my instructor, I realized that keeping the dashboard on a single page helps users understand the data more quickly. This pushed me to remove less important information and focus more on clarity, hierarchy, and usability. I also practiced using AI as a learning tool, such as generating ideas, checking logic, and refining design decisions, which helped me work more efficiently.

3

Evaluate at the end of class:

1. What did I learn?
2. What stuck?
3. What’s next?

Through this project, I learned how to design dashboards with a stronger focus on visual hierarchy, data prioritization, and user understanding. By practicing repeatedly, I became more confident using Figma and combining design skills with AI support. I also turned my final design into a working webpage, which helped me understand how design decisions translate into a real interface. This project helped me see how iteration and feedback improve both design quality and my own learning process. My next step is to continue refining the visuals and interactions, and the final dashboard can be viewed here:

https://guoyund.phoenix.sheridanc.on.ca/The-Daily-Grind-Dashboard-main/