Personal Grocery Inflation Analysis (March 2023 – May 2025)

A micro-level inflation model using real household receipts to expose behavioral and structural price signals missed by national CPI.

This project investigates inflation as it is actually experienced by a household, using over 600 structured grocery receipts from Albert Heijn (March 2023 – May 2025). Rather than summarizing spending, it builds a **behaviorally aware, structurally grounded price index**, then benchmarks it against the CBS national CPI for food and non-alcoholic beverages.

The methodology mixes classical CPI logic (Laspeyres indexing) with psychological and economic insights from consumer behavior research. The goal: reconstruct **perceived and actual inflation** from raw, real-world purchase data, not estimated categories.

Why Personal CPI?

National CPI figures are macroeconomic averages. They miss heterogeneity.
Household inflation varies significantly (Kiss & Strasser, 2022), not due to different needs, but due to behavioral habits: brand loyalty, discount timing, basket rigidity.

We use structured receipt data to replicate a household-level CPI and track divergence between:

- **What's measured nationally**, and
- **What's experienced personally**.

> All item prices reflect **listed shelf prices, excluding discounts**. This isolates structural price shifts from shopping strategy, ensuring comparability across time.

Indicators Computed

We construct three inflation measures:

1. **Expenditure-Weighted Laspeyres**

    - Tracks the cost of a fixed April 2023 basket over time.
    - Reflects **financial exposure** to price shifts.
    - Mirrors official CPI methodology.

2. **Frequency-Weighted Laspeyres**

   - Uses the same basket, but weights by unit count.
   - Emphasizes **psychological salience**: frequently seen = more inflation noticed.
   - Motivated by D'Acunto et al. (2021).

3. **Behavioral Proxies**

   - Tracks total monthly **grocery spending**, normalized to April 2023 = 100.
   - Captures substitution, seasonality, and bulk-buying effects.
   - Not a price index, but a **cost-of-living proxy**.

All indices are normalized to April 2023 = 100.

Benchmark: CBS CPI for Food and Non-Alcoholic Drinks

As an external reference, we use CBS StatLine Table 83131NED.
This tracks national-level food inflation and serves as a **macro benchmark**. Unlike our household indices, it uses plutocratic weighting and category-level aggregation.

Data Sources

- **Receipt Data**: Parsed from 600+ Albert Heijn digital receipts (5,000+ product lines), spanning March 2023 – May 2025. All prices reflect **pre-discount shelf values** to isolate structural trends. (Tait, V. 2025)

- **CBS Benchmark**: Official CPI data for food/non-alcoholic beverages retrieved from CBS StatLine, June 2025. (CBS, 2025)

Receipt Data Preprocessing

We begin by loading the structured grocery dataset generated from over 600 digitized Albert Heijn receipts. Timestamps are parsed into datetime format, and monthly periods are extracted for CPI-style aggregation.

We apply a minimal filter to remove any rows with non-positive prices. Additional filters (e.g., on product names or units) were tested but found to have no effect and were left commented for clarity. A manual spot check confirmed that only 2 malformed rows were dropped (5105 → 5103), and no timestamp parsing errors were found.

Each product line includes a total price and a unit count. We derive per-unit prices accordingly, which will later be used to compute both frequency-weighted and expenditure-weighted inflation indices.

> **Important:** All prices used in this analysis reflect **pre-discount shelf prices**, as extracted directly from the receipts. Any temporary Bonus or Bonusbox reductions are **excluded by design**. This ensures that all indices track structural price levels, not transient promotional effects.

> **Note on Coverage:** While March 2023 data is present, it is incomplete. The first full month with reliable coverage is **April 2023**, which serves as the reference point for all subsequent indexing and basket construction.

In [13]:

```python
# ----------------------------------------
# Load and Preprocess Receipt Data
# ----------------------------------------

import pandas as pd

# Load the structured receipt data
receipt_data = pd.read_csv(
    "data/personal/all_receipts.csv",
    sep="|"
)

# Parse timestamp column to datetime
receipt_data["timestamp"] = pd.to_datetime(
    receipt_data["timestamp"],
    errors="coerce"
)

# Derive per-unit price
receipt_data["unit_price"] = (
    receipt_data["price_total"] /
    receipt_data["units"]
)

# Filter out malformed or nonsense rows
# Only 2 rows dropped (5105 → 5103),
# solely due to price_total ≤ 0.
# product_name and units filters were tested
# but had no effect, left commented for reference.
receipt_data = receipt_data[
    (receipt_data["price_total"] > 0)
    # & (receipt_data["product_name"].notna())
    # & (receipt_data["units"] > 0)
]

# Warning in case any timestamps failed to parse
assert receipt_data["timestamp"].notna().all(), (
    "Warning: Some timestamps failed to parse."
)

# Extract monthly period for CPI-style aggregation
receipt_data["month"] = (
```

```
    receipt_data["timestamp"].dt.to_period("M")
)

# Preview (commented to preserve privacy)
# receipt_data.head()
```

Processing the CBS CPI Benchmark

To benchmark personal inflation against official figures, we use the Dutch **Consumer Price Index (CPI)** for *Voedingsmiddelen en niet-alcoholische dranken*, the closest available match to the author's grocery spending category.

This data was downloaded from CBS StatLine (table 83131NED) and covers monthly index values from March 2023 to May 2025, with 2015 as the base year (2015 = 100). Since the source file uses Dutch month names (e.g., *"2023 april"*), we map these to standard date formats for proper alignment.

We retain only the cleaned columns needed for analysis:

- `month` : The reporting period as `YYYY-MM`
- `cbs_cpi` : The official CPI index level for that month

This will later be compared directly against the personal CPI series to visualize divergence over time.

> 🔍 **Note:** All official CPI figures are rebased to **April 2023 = 100** for consistency with the personal CPI baseline.

In [14]:
```
# ----------------------------------------
# Load and Preprocess CBS CPI Benchmark
# ----------------------------------------

# Load CBS CPI data
cbs_data = pd.read_csv(
    "data/cbs/CBS_CPI_Food&Non-Alcoholic_2015_index_cleaned.csv",
    sep=";"
)

# Define Dutch month mapping
dutch_months = {
    "januari": "01",
    "februari": "02",
    "maart": "03",
    "april": "04",
    "mei": "05",
    "juni": "06",
    "juli": "07",
    "augustus": "08",
    "september": "09",
    "oktober": "10",
```

```python
    "november": "11",
    "december": "12"
}

# Extract year and month from 'Perioden'
cbs_data[["year", "month_name"]] = (
    cbs_data["Perioden"]
    .str.extract(r"(\d{4})\s+([a-z]+)", expand=True)
)
cbs_data["month_num"] = (
    cbs_data["month_name"].map(dutch_months)
)
cbs_data["date"] = pd.to_datetime(
    cbs_data["year"] + "-" +
    cbs_data["month_num"] + "-01"
)

# Rename CPI column
cbs_data = cbs_data.rename(
    columns={"CPI 2015 Index": "cbs_cpi"}
)

# Convert CPI column to numeric
cbs_data["cbs_cpi"] = (
    cbs_data["cbs_cpi"]
    .str.replace(",", ".")
    .astype(float)
)

# Drop rows with missing CPI
cbs_data = cbs_data.dropna(
    subset=["cbs_cpi"]
)

# Extract monthly period
cbs_data["month"] = (
    cbs_data["date"].dt.to_period("M")
)

# Keep final columns
cbs_data = cbs_data[["month", "cbs_cpi"]].reset_index(
    drop=True
)

# ----------------------------------------
# Rebase CBS CPI to March 2023 = 100
# ----------------------------------------

# Get base value from April 2023
base_cbs_value = cbs_data.loc[
    cbs_data["month"] == "2023-04",
    "cbs_cpi"
].values[0]

# Rebase to April 2023 = 100
cbs_data["cbs_cpi_rebased"] = (
```

```
    100 * cbs_data["cbs_cpi"] / base_cbs_value
)

# Drop CBS data before base month
cbs_data = cbs_data[
    cbs_data["month"] >= base_month
]

# Preview
cbs_data.head(3)
```

Out[14]:

	month	cbs_cpi	cbs_cpi_rebased
1	2023-04	136.70	100.000000
2	2023-05	137.83	100.826628
3	2023-06	138.22	101.111924

Basket Construction: Fixed Prices vs. Real Behavior

We construct three inflation indicators to reflect different perspectives on experienced grocery price dynamics:

---

1. Laspeyres Index – Expenditure Weighted

This index tracks how the cost of a fixed consumption basket evolves over time. The basket is constructed from all products purchased in **April 2023**, with quantities and composition held constant. Product weights are proportional to their share of total April expenditure, emphasizing **financial exposure**.

Mathematically:

- Let p_{it} be the price of product *i* in month *t*
- Let q_{i0} be the quantity bought in **April 2023**
- Index:
  $$L_t^{\text{exp}} = \frac{\sum_i p_{it} \cdot q_{i0}}{\sum_i p_{i0} \cdot q_{i0}} \cdot 100$$

This mirrors official CPI methodology (IMF, 2004) and tends to **overstate inflation**, as it assumes no substitution.

---

2. Laspeyres Index – Frequency Weighted

Also based on the **April 2023** basket, this variant replaces expenditure weights with **unit frequency weights**. This emphasizes **salience**: commonly purchased products receive more weight, regardless of price.

Mathematically:

- Weight for product *i*:
  [w_i^{\text{freq}} = \frac{q_{i0}}{\sum_j q_{j0}}]
- Index:
  [L_t^{\text{freq}} = \sum_i w_i^{\text{freq}} \cdot \frac{p_{it}}{p_{i0}} \cdot 100]

This version is useful for modeling **perceived inflation**, which behavioral studies suggest skews toward salient or frequently encountered prices (D’Acunto et al., 2021).

---

3. **Behavioral Proxy – Total Expenditure Index**

Rather than trying to reprice dynamic monthly baskets (which breaks under sparse base prices), we track **total grocery expenditure per month**, indexed to **April 2023 = 100**.

This proxy reflects:

- Substitution behavior (e.g., toward cheaper brands)
- Promotional effects
- Seasonal shifts in basket composition
- Quantity changes

It is not a true price index, but provides a **behaviorally grounded signal** of changing grocery cost burden.

Final Index Set

Indicator	Construction Basis	Weighting	Interpretation
Laspeyres – Expenditure	April 2023 basket	Money spent	Financial exposure, CPI-like
Laspeyres – Frequency	April 2023 basket	Unit count	Perceived inflation (salience-based)
Behavioral Proxy – Expenditure	Monthly total spend	None	Adaptive cost signal, includes substitution

All three indices are **normalized to April 2023 = 100**.

Laspeyres Index – Expenditure Weighted

We begin by computing the **Laspeyres index** using **expenditure weights**.

This index tracks how the prices of goods bought in **April 2023** evolve over time, under the assumption that the consumption basket remains fixed. Product quantities and product composition are locked to their April levels; only prices are allowed to vary.

Each product's contribution to the index is weighted by its **share of total expenditure** in April 2023. This reflects financial exposure: products that consumed a larger share of the grocery budget in April receive greater influence on the index trajectory.

This version closely mirrors **official CPI methodology** (IMF, 2004) and serves as the benchmark for comparison against adaptive indices.

> This index **overstates inflation** if substitution occurs, as it assumes the consumer keeps buying the same expensive items despite price increases.

In [15]:
```
# ----------------------------------------
# Laspeyres Index - Expenditure Weighted
# ----------------------------------------

# Define base period
base_month = "2023-04"

# Filter static basket from April 2023
april_data = receipt_data[
    receipt_data["month"] == base_month
].copy()

# Aggregate total quantity and expenditure per product
static_basket_exp = (
    april_data
    .groupby("product_name")
    .agg(
        q_april=("units", "sum"),
        p_april=("unit_price", "mean")
    )
    .reset_index()
)

# Drop any products with zero units
static_basket_exp = static_basket_exp[
    static_basket_exp["q_april"] > 0
]

# Compute expenditure weights from April 2023
static_basket_exp["expenditure_april"] = (
    static_basket_exp["q_april"] *
    static_basket_exp["p_april"]
)
static_basket_exp["weight_exp"] = (
    static_basket_exp["expenditure_april"] /
    static_basket_exp["expenditure_april"].sum()
)

# Get average monthly prices from April 2023 onward
```

```python
monthly_prices_exp = receipt_data[
    receipt_data["month"] >= base_month
].copy()
monthly_avg_prices_exp = (
    monthly_prices_exp
    .groupby(["month", "product_name"])
    .agg(p_t=("unit_price", "mean"))
    .reset_index()
)

# Create month × product grid
months_exp = monthly_avg_prices_exp["month"].unique()
product_month_grid_exp = pd.MultiIndex.from_product(
    [months_exp, static_basket_exp["product_name"]],
    names=["month", "product_name"]
).to_frame(index=False)

# Merge prices and static basket info
laspeyres_exp_data = (
    product_month_grid_exp
    .merge(monthly_avg_prices_exp, on=["month", "product_name"], how="left")
    .merge(static_basket_exp, on="product_name", how="left")
)

# Fill missing prices with base price
laspeyres_exp_data["p_t"] = laspeyres_exp_data["p_t"].fillna(
    laspeyres_exp_data["p_april"]
)

# Calculate price relatives and contributions
laspeyres_exp_data["price_relative"] = (
    laspeyres_exp_data["p_t"] / laspeyres_exp_data["p_april"]
)
laspeyres_exp_data["contrib"] = (
    laspeyres_exp_data["price_relative"] *
    laspeyres_exp_data["weight_exp"]
)

# Aggregate monthly index
laspeyres_exp_index = (
    laspeyres_exp_data
    .groupby("month")["contrib"]
    .sum()
    .reset_index()
)
laspeyres_exp_index = laspeyres_exp_index.rename(
    columns={"contrib": "laspeyres_exp"}
)
laspeyres_exp_index["laspeyres_exp"] *= 100  # Normalize to base = 100

# Preview
laspeyres_exp_index.head(3)
```

Out[15]:

	month	laspeyres_exp
0	2023-04	100.000000
1	2023-05	101.704958
2	2023-06	100.033936

Laspeyres Index – Frequency Weighted

We now compute the **Laspeyres index** using **frequency-based weights**, an alternative to financial weighting.

The basket remains fixed based on **April 2023**, but instead of using expenditure shares, we weight products by their **purchase frequency**, i.e., how many units were bought. This means that commonly bought (and likely more salient) products carry more weight, regardless of their price.

This approach aligns with behavioral evidence on **perceived inflation**: consumers tend to overweight frequently purchased goods when forming expectations (D'Acunto et al., 2021), even if those goods represent a small part of their total spending.

> While the expenditure-weighted Laspeyres reflects **economic burden**, this version emphasizes **consumer attention** and psychological salience.

In [16]:

```python
# ----------------------------------------
# Laspeyres Index - Frequency Weighted
# ----------------------------------------

# Filter static basket from April 2023
april_data_freq = receipt_data[
    receipt_data["month"] == base_month
].copy()

# Aggregate total quantity per product (no prices used here)
static_basket_freq = (
    april_data_freq
    .groupby("product_name")
    .agg(
        q_april=("units", "sum"),
        p_april=("unit_price", "mean")  # still needed for later price relatives
    )
    .reset_index()
)

# Drop any products with zero units
static_basket_freq = static_basket_freq[
    static_basket_freq["q_april"] > 0
]
```

```python
# Compute frequency weights (pure unit share)
static_basket_freq["weight_freq"] = (
    static_basket_freq["q_april"] /
    static_basket_freq["q_april"].sum()
)

# Get average monthly prices from April 2023 onward
monthly_prices_freq = receipt_data[
    receipt_data["month"] >= base_month
].copy()
monthly_avg_prices_freq = (
    monthly_prices_freq
    .groupby(["month", "product_name"])
    .agg(p_t=("unit_price", "mean"))
    .reset_index()
)

# Create full month × product grid
months_freq = monthly_avg_prices_freq["month"].unique()
product_month_grid_freq = pd.MultiIndex.from_product(
    [months_freq, static_basket_freq["product_name"]],
    names=["month", "product_name"]
).to_frame(index=False)

# Merge in prices and static basket info
laspeyres_freq_data = (
    product_month_grid_freq
    .merge(monthly_avg_prices_freq, on=["month", "product_name"], how="left")
    .merge(static_basket_freq, on="product_name", how="left")
)

# Fill missing prices with April price
laspeyres_freq_data["p_t"] = laspeyres_freq_data["p_t"].fillna(
    laspeyres_freq_data["p_april"]
)

# Compute price relatives and frequency-weighted contribution
laspeyres_freq_data["price_relative"] = (
    laspeyres_freq_data["p_t"] /
    laspeyres_freq_data["p_april"]
)
laspeyres_freq_data["contrib"] = (
    laspeyres_freq_data["price_relative"] *
    laspeyres_freq_data["weight_freq"]
)

# Aggregate monthly index
laspeyres_freq_index = (
    laspeyres_freq_data
    .groupby("month")["contrib"]
    .sum()
    .reset_index()
)
laspeyres_freq_index = laspeyres_freq_index.rename(
    columns={"contrib": "laspeyres_freq"}
)
```

```
laspeyres_freq_index["laspeyres_freq"] *= 100  # Normalize to base = 100

# Preview
laspeyres_freq_index.head(3)
```

Out[16]:

	month	laspeyres_freq
0	2023-04	100.000000
1	2023-05	103.516692
2	2023-06	101.065685

Behavioral Proxy – Total Expenditure Index

Finally, we compute a **behavioral proxy** for inflation based on **total monthly grocery spending**, normalized to **April 2023 = 100**.

Unlike the Laspeyres indices, this method does not rely on fixed product-level quantities or explicit price comparisons. Instead, it directly tracks how much money was actually spent each month, capturing the evolving **real-world cost of feeding a household**.

This index reflects a number of behavioral and economic effects, including:

- Substitution toward cheaper or discounted products
- Reactions to price promotions
- Seasonal fluctuations in consumption
- Changes in overall quantity purchased

While this approach cannot isolate price changes from volume changes, it **captures the consumer's actual cost burden**, which is arguably the most practical measure of experienced inflation.

> This proxy **undershoots pure inflation** if substitution or promotional responsiveness increases over time, but it reflects what the household truly paid, no assumptions, no price imputation.

In [17]:

```
# ----------------------------------------
# Behavioral Proxy Index - Expenditure
# ----------------------------------------

# Compute total monthly grocery expenditure
monthly_expenditure = (
    receipt_data
    .groupby("month")
    .agg(total_expenditure=("price_total", "sum"))
    .reset_index()
)
```

```
# Drop incomplete early months (e.g., March 2023 or earlier)
monthly_expenditure = monthly_expenditure[
    monthly_expenditure["month"] >= base_month
]

# Normalize to April 2023 = 100
base_value = monthly_expenditure.loc[
    monthly_expenditure["month"] == base_month,
    "total_expenditure"
].values[0]

monthly_expenditure["behavioral_exp"] = (
    100 * monthly_expenditure["total_expenditure"] / base_value
)

# Keep only month and index
behavioral_exp_index = monthly_expenditure[[
    "month", "behavioral_exp"
]]

# Preview
behavioral_exp_index.head(3)
```

Out[17]:

	month	behavioral_exp
2	2023-04	100.000000
3	2023-05	148.806330
4	2023-06	172.472578

Visualization of Inflation Indices

We now generate plots to visualize the behavior of the computed personal inflation indices, alongside the official CBS CPI benchmark.

Two types of plots are produced:

1. **Indexed Levels Over Time**, showing the overall inflation trajectory.
2. **Month-over-Month Changes**, highlighting volatility, seasonality, and behavioral shifts.

> All series are normalized to **April 2023 = 100** for consistent baseline comparison.

In [9]:

```
# ----------------------------------------
# Plot: Indexed Inflation Measures
# ----------------------------------------

import matplotlib.pyplot as plt

# Merge all index series into a single DataFrame
all_indices = (
```

```python
    laspeyres_exp_index
    .merge(laspeyres_freq_index, on="month", how="outer")
    .merge(behavioral_exp_index, on="month", how="outer")
    .merge(
        cbs_data[["month", "cbs_cpi_rebased"]],
        on="month",
        how="outer"
    )
)

# Rename for clarity
all_indices = all_indices.rename(columns={
    "laspeyres_exp": "Laspeyres (Expenditure)",
    "laspeyres_freq": "Laspeyres (Frequency)",
    "cbs_cpi_rebased": "CBS Food and Non-Alcoholic Drink CPI",
    "behavioral_exp": "Total Indexed Expenditure"
}).sort_values("month")

# Plot the indices
plt.figure(figsize=(12, 6))

for col in [
    "Laspeyres (Expenditure)",
    "Laspeyres (Frequency)",
    "CBS Food and Non-Alcoholic Drink CPI",
    "Total Indexed Expenditure"
]:
    plt.plot(
        all_indices["month"].astype(str),
        all_indices[col],
        label=col
    )

plt.axhline(100, color="gray", linestyle="--", linewidth=0.8)
plt.title("Figure 1: Indexed Inflation Measures (April 2023 = 100)")
plt.xlabel("Month")
plt.ylabel("Index Level")
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.grid(True)
plt.show()
```

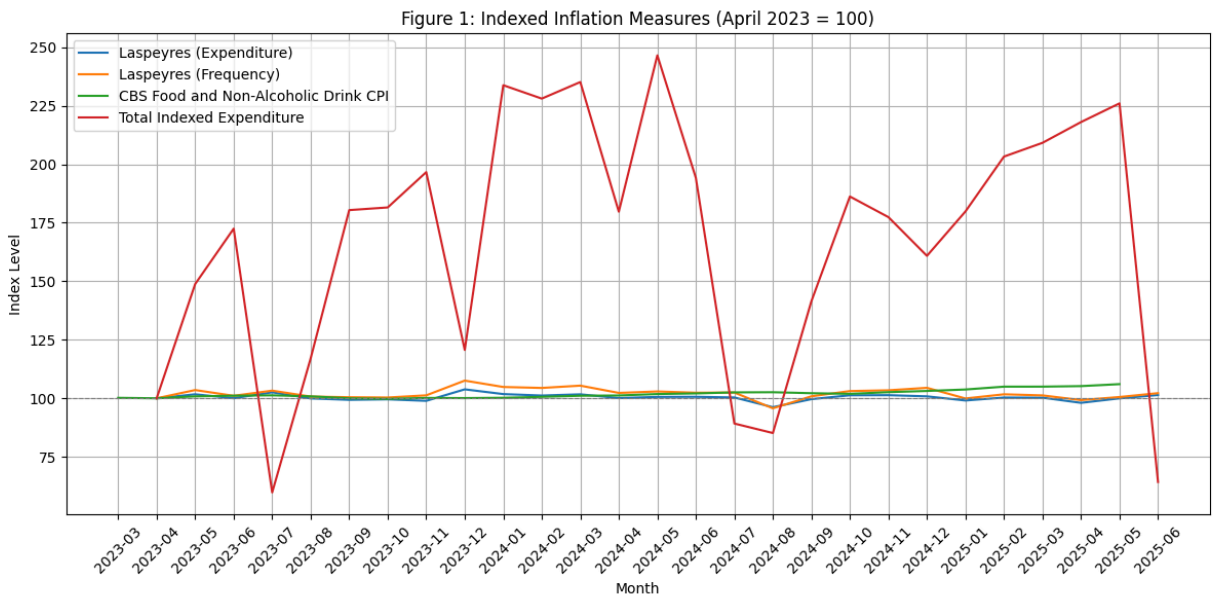

Figure 1 presents four indexed inflation series: the expenditure- and frequency-weighted Laspeyres indices, a behavioral proxy based on total monthly grocery expenditure, and the CBS CPI for food and non-alcoholic beverages.

The **Total Indexed Expenditure** series exhibits substantial volatility and pronounced seasonal patterns, in contrast to the more stable trajectories of the Laspeyres and CBS indices. This behavior stems from the fact that total monthly spending captures not just price movements, but also variation in purchasing behavior, including periods where grocery spending is absent or suppressed due to travel or temporary reliance on other households (e.g., staying with family).

Importantly, the level and shape of this series are highly sensitive to the choice of base month. Because this proxy includes periods with zero or reduced expenditure, its index can shift significantly based on whether the base month coincides with a high- or low-spending period.

Given its behavioral volatility and limited comparability to price-based indices, we exclude **Total Indexed Expenditure** from subsequent plots. The remaining visualizations focus on more stable measures of price change to better illustrate inflation dynamics.

In [21]:
```
# ----------------------------------------
# Plot: Indexed Inflation Measures (Figure 2 – Expenditure Proxy Removed)
# ----------------------------------------
# Reuse the same merged index DataFrame
# Already contains all indices, no need to remerge

# Plot only the stable indices
plt.figure(figsize=(12, 6))

for col in [
    "Laspeyres (Expenditure)",
    "Laspeyres (Frequency)",
```

```
    "CBS Food and Non-Alcoholic Drink CPI"
]:
    plt.plot(
        all_indices["month"].astype(str),
        all_indices[col],
        label=col
    )

plt.axhline(100, color="gray", linestyle="--", linewidth=0.8)
plt.title("Figure 2: Indexed Inflation Measures Without Expenditure Proxy (April 20
plt.xlabel("Month")
plt.ylabel("Index Level")
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.grid(True)
plt.show()
```

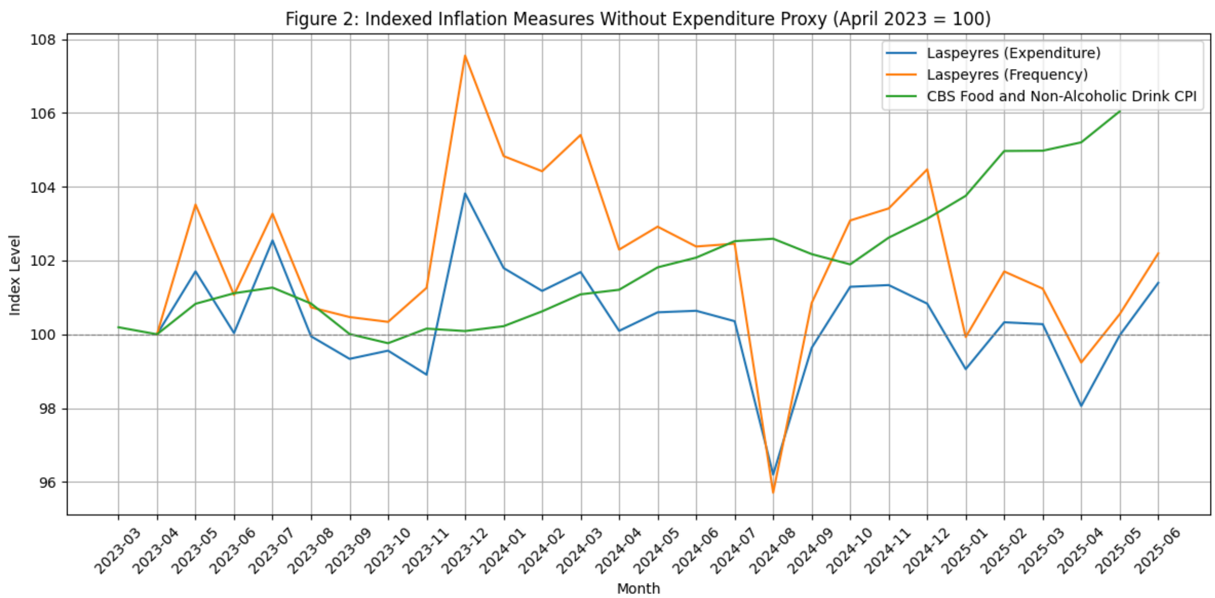

Figure 2 mirrors the structure of Figure 1, but excludes the volatile expenditure proxy to better highlight relative differences between the remaining series. All indices are normalized to **April 2023 = 100**.

The two Laspeyres indices remain strongly correlated, as expected given their shared static basket and only differing weighting schemes. However, their short-term fluctuations differ due to variations in purchase frequency vs. financial weighting, making both useful for different inflation perspectives.

A sharp dip in **August 2024** appears in both personal indices, while the CBS CPI maintains a smooth upward trajectory. Since all prices are **pre-discount shelf prices**, this decline likely reflects a genuine drop in listed prices for one or more **heavily weighted April 2023 products**.

Potential causes include:

- Temporary price corrections or price wars
- Supplier-driven markdowns
- Systematic price restructuring at Albert Heijn

This divergence highlights the value of personal inflation indices for detecting real-world price dynamics that are **invisible in national aggregates**.

Inflation Rate: Month-over-Month Changes

Next, we plot the **inflation rate**, defined as the month-over-month percentage change in each index.

While level indices show long-term price drift, inflation rates provide a clearer picture of **short-term volatility**, **momentum**, and **seasonal effects**. They also help distinguish between periods of stable inflation and those with sharp shifts, whether driven by genuine price changes or behavioral anomalies.

Tracking inflation this way reveals how sensitive a fixed basket is to underlying price dynamics and helps identify turning points that are often smoothed out in cumulative index plots.

In [22]:

```
# ----------------------------------------
# Plot: Month-over-Month Inflation Rates
# ----------------------------------------

# Create a copy to avoid modifying original index levels
inflation_rates = all_indices.copy()

# Compute discrete MoM % changes for each index
for col in [
    "Laspeyres (Expenditure)",
    "Laspeyres (Frequency)",
    "CBS Food and Non-Alcoholic Drink CPI"
]:
    inflation_rates[f"{col} MoM"] = (
        inflation_rates[col].pct_change(fill_method=None) * 100
    )

# Plot month-over-month inflation rates

plt.figure(figsize=(12, 6))

for col in [
    "Laspeyres (Expenditure) MoM",
    "Laspeyres (Frequency) MoM",
    "CBS Food and Non-Alcoholic Drink CPI MoM"
]:
    plt.plot(
        inflation_rates["month"].astype(str),
        inflation_rates[col],
```

```
        label=col
    )

plt.axhline(0, color="gray", linestyle="--", linewidth=0.8)
plt.title("Figure 3: Month-over-Month Inflation Rates")
plt.xlabel("Month")
plt.ylabel("MoM % Change")
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.grid(True)
plt.show()
```

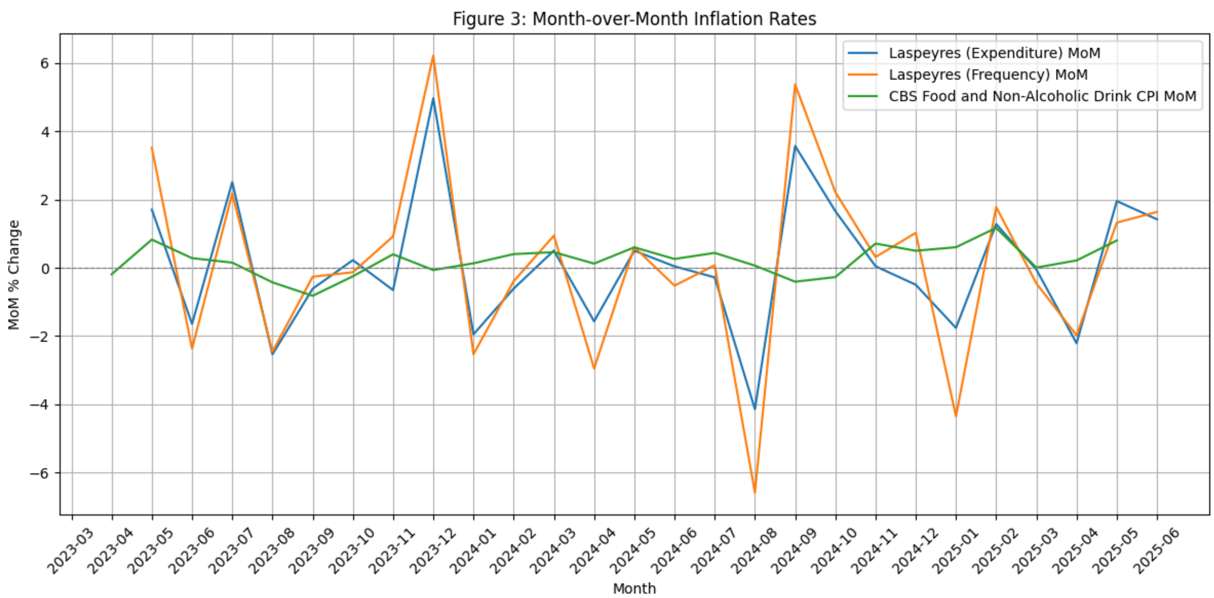

Figure 3 presents the **month-over-month inflation rates** for each index, calculated as the percentage change from the previous month.

The CBS CPI displays a smooth, low-volatility trajectory, consistent with its role as a national average across thousands of products and households. In contrast, the personal Laspeyres indices, especially the **frequency-weighted variant**, exhibit much greater short-term volatility, with swings ranging from –6% to +6%.

This increased noise reflects two structural factors:

- A **narrow, personalized basket** with limited product diversification
- A **fixed weighting scheme**, which magnifies the impact of large price movements in any heavily weighted item

A striking anomaly occurs in **August 2024**, where both personal indices show a sharp drop not mirrored in the CBS benchmark. This suggests a genuine price correction in one or more salient household products, likely unimportant at the national level. Even more curiously, this drop is **nearly fully reversed in September**, hinting at a transient distortion such as a temporary markdown, supply-driven clearance, or even a one-off pricing artifact.

> Overall, plotting inflation rates, rather than just index levels, reveals important dynamics: sudden shocks, short-lived anomalies, and behavioral noise become visible in ways that cumulative indices simply obscure.

Conclusion

This project constructed a personal CPI not simply to summarize grocery spending, but to expose **structural and behavioral price signals** that standard inflation metrics often miss. By applying classical CPI techniques to real receipt data we highlight how inflation is actually felt, not just measured.

Using over 600 receipts spanning March 2023 to May 2025, the project implemented both expenditure- and frequency-weighted Laspeyres indices as well as a behavioral proxy based on total monthly spending. These were benchmarked against CBS national figures for food and non-alcoholic drinks to reveal areas of overlap, divergence, and hidden volatility.

Key findings:

- The **expenditure-weighted Laspeyres index** tracked the CBS benchmark closely but showed greater short-term variability, reflecting personalized price exposure.
- The **frequency-weighted index** exhibited higher volatility still, consistent with behavioral research showing that consumers overweight salient, frequently purchased items.
- A pronounced **drop in August 2024**, observed in both personal indices but not in the CBS CPI, points to a structural price shock in a household-relevant item, followed by an immediate rebound in September.
- The **month-over-month inflation rate plot** revealed dynamics (e.g., temporary corrections, spikes, and reversals) that would have remained hidden in cumulative index levels.

Together, these results demonstrate how inflation can differ materially from the national average when measured at the household level, even using a simple, transparent methodology.

Discussion

Several deliberate modeling choices shaped this analysis, each with strengths and limitations.

- **Pre-discount shelf prices** were used to track structural price changes, independent of short-term promotional behavior. This makes the indices more stable and interpretable, but may overstate experienced inflation when discounts are heavily used.

- A **fixed basket from April 2023** was chosen for transparency and comparability. However, this introduces **substitution bias**: the index assumes the same products are bought every month, even when prices change. A chained or rolling basket could address this in future work, but proved to unstable this time.

- The choice of **Laspeyres indices** reflects CPI orthodoxy, but other formulations, such as Paasche or Fisher, were ruled out due to sparse and inconsistent base-price matching in real receipt data.

- The **behavioral proxy** based on total expenditure captures lived spending pressure, but blends price and quantity changes, making it hard to isolate true inflation effects.

- Finally, using a **single-household dataset** offers clarity and personal relevance, but limits generalizability. These results highlight intra-household volatility more than macro trends.

Despite these constraints, the results are robust and behaviorally grounded. The volatility seen in month-over-month inflation rates, and the divergence from CBS figures in August 2024, underscore the value of **personal-level inflation tracking**. These deviations are not just noise; they represent genuine, often overlooked, consumer experience.

References

D'Acunto, F., Malmendier, U., Ospina, J., & Weber, M. (2021). Exposure to grocery prices and inflation expectations. *Journal of Political Economy, 129*(5). https://doi.org/10.1086/713192

European Central Bank (Kiss, R., & Strasser, G.). (2022). *Heterogeneity in inflation expectations and behavior* (ECB Working Paper No. 2898). https://www.ecb.europa.eu/pub/pdf/scpwps/ecb.wp2898~29405f932f.en.pdf

International Monetary Fund. (2004). *Consumer price index manual: Theory and practice*. https://doi.org/10.5089/9789221136996.069

Kaplan, G., & Schulhofer-Wohl, S. (2017). Inflation at the household level. *Journal of Monetary Economics, 91*, 19–38. https://doi.org/10.1016/j.jmoneco.2017.08.002

Statistics Netherlands (CBS). (2025). *Consumentenprijsindex (CPI) 2015=100, vanaf 1996* [Data set]. CBS StatLine. https://opendata.cbs.nl/statline/#/CBS/nl/dataset/83131NED/table *Data retrieved on 19 June 2025, covering March 2023 to May 2025.*

Tait, V. (2025). Albert Heijn Receipt Parser [Python code]. GitHub. https://github.com/vincenttait/ah-receipt-parser (Used to generate structured grocery data from scanned PDF receipts, March 2023 – May 2025.)