Introduction and System Design

The project is a web-based supermarket simulation that lets users create transactions and mine association rules using Apriori, Eclat, and FP-Growth. It's built with Next.js, React, TypeScript, Zustand, and Tailwind, following a clean, modular architecture. Users browse products, build carts, generate transactions, and view mined patterns through an interactive interface.

Data Preprocessing Approach

A preprocessing pipeline cleans all incoming transactions before mining. It removes empty and single-item transactions, deduplicates items, standardizes names (lowercase/trim), and filters invalid products based on a predefined catalog. The system tracks statistics for every correction to ensure rule mining uses only valid, consistent data.

Algorithm Implementation Details (with pseudocode)

Apriori

Level-wise algorithm that generates candidate itemsets and prunes by minimum support.
Pseudocode:

```
L1 = frequent 1-itemsets
for k = 2..n:
  Ck = generate candidates from Lk-1
  count support of each candidate
  Lk = candidates with support ≥ min_sup
  if Lk empty: break
```

Eclat

Uses vertical TID-sets and depth-first search to compute itemset intersections.
Pseudocode:

```
Build TID-set for each item
DFS(prefix, TIDs):
  for each remaining item:
    newTID = TID(prefix) ∩ TID(item)
    if support ≥ min_sup:
      DFS(prefix + item, newTID)
```

FP-Growth

Builds an FP-Tree and mines conditional pattern bases without generating candidates.
Pseudocode:

```
Scan DB → count items
```

```
Sort items → build FP-tree
for each item in header:
  construct conditional pattern base
  build conditional FP-tree
  mine recursively
```

Performance Analysis and Comparison

All three algorithms were tested on a 100-transaction dataset with 30 products. FP-Growth was fastest (≈8–18ms), followed by Eclat (≈10–20ms), and Apriori was slowest (≈15–25ms). While all find the same rules, FP-Growth scales best, Apriori slows significantly as data grows, and Eclat performs well but uses more memory on dense data.

User Interface Design

The UI uses glassmorphism styling, responsive layouts, and clear visual hierarchy to make data mining approachable for non-technical users. Users can browse products, adjust carts, import transactions, view preprocessing feedback, run mining algorithms, and explore rules. Smooth animations, accessibility features, and mobile-first layouts ensure a clean, modern experience.

Testing and Results

Testing covered product browsing, transaction creation, CSV imports, preprocessing, and rule mining. All three algorithms produced identical frequent itemsets and rules, confirming correctness. Preprocessing successfully removed empty/single transactions, duplicates, and invalid items. The UI responded quickly, mining stayed under 50ms, and the system handled edge cases like malformed CSVs gracefully.

Conclusion and Reflection

The project successfully demonstrates how association rule mining works in a realistic supermarket context, packaged in a modern and approachable web interface. Building it reinforced the importance of preprocessing, algorithm efficiency, and usability—especially when running complex logic client-side. Implementing Apriori, Eclat, and FP-Growth deepened understanding of their strengths and trade-offs, while UI design choices helped make the data mining process intuitive. Overall, the project achieved its goals and provides a strong foundation for future enhancements like server-side processing, advanced visualizations, or real-world retail applications.

Students:

- Carlos Mejia
- Mandy Saint Simon
- Pablo Valdes