SCHOOL OF INFORMATION & SYSTEMS TECHNOLOGY

BeiRadar: A Web-Based Supermarket Price Comparison System

Submitted by

LILIAN WANJIRU NG'ANG'A

668828

In Partial Fulfilment For The Award Of Bachelor of Science Degree

In

APPLIED COMPUTER TECHNOLOGY(APT)

FALL 2025

APT 3065: MID/FINAL TERM PROJECT

DECLARATION

I declare that this is my original work and has never been submitted to any institution for the Certificate, Diploma or Degree award. This document has been submitted to fulfil the requirements for a Degree at USIU University.

STUDENT: LILIAN WANJIRU NG'ANG'A

Signature:…….. Date:……..

This project has been submitted for examination purposes with the approval of the supervisor

SUPERVISOR: DR. FREDRICK OGORE

Signature:…… Date:……

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to Dr. Fredrick Ogore for his invaluable guidance, support, and constructive feedback throughout the development of this project. His expertise and insights significantly contributed to the successful completion of BeiRadar.

I am also grateful to my family and friends for their encouragement and moral support during this academic journey. Their patience and understanding made it possible for me to dedicate time and effort to this project.

Finally, I acknowledge the resources and facilities provided by the institution, which were instrumental in the research, design, and implementation phases of this project.

DEDICATION

This project is dedicated to my family, whose unwavering support and encouragement have been my foundation throughout my academic journey. Their belief in my abilities has been a constant source of motivation.

I also dedicate this work to all consumers in Kenya who struggle with rising living costs and lack access to transparent pricing information. May this system contribute to making informed shopping decisions more accessible to all.

ABSTRACT

The rising cost of living in Kenya, particularly driven by food inflation, has created significant financial strain on households. Despite the proliferation of online shopping platforms, consumers lack a centralized tool to compare product prices across multiple supermarkets, leading to overspending and inefficient budgeting. This project presents BeiRadar, a web-based supermarket price comparison system designed to address this gap by aggregating and displaying product prices from major Nairobi supermarkets including Carrefour, Naivas, and Quickmart.

BeiRadar employs a three-tier architecture comprising a presentation layer (HTML, CSS, JavaScript), an application layer (Python Flask), and a data layer (SQLite database). The system integrates hybrid data collection methods combining automated web scraping with manual data entry to ensure comprehensive coverage and accuracy. Key features include product search and filtering, shopping cart functionality with multi-store cost comparison, real-time price updates, and deals/promotions tracking.

The implementation utilized Python libraries including Flask for backend development, BeautifulSoup for web scraping, and Pandas for data processing. The database stores product information, prices, categories, and images, with automated matching algorithms linking products to their visual representations. The user interface provides intuitive navigation through categories, subcategories, and advanced filtering options including price range and discount percentages.

Testing revealed that BeiRadar successfully enables users to identify the most cost-effective shopping options, with the cart comparison feature calculating total costs across all supermarkets simultaneously. The system demonstrated reliability in handling concurrent user queries and maintaining data accuracy through regular updates.

BeiRadar represents a significant contribution to Kenya's digital economy by providing localized consumer empowerment tools. The platform promotes price transparency, encourages retail competition, and supports household financial planning during periods of economic pressure. Future enhancements could include API integration for real-time data updates, mobile application development, and expansion to additional supermarkets and cities.

Keywords: Price comparison, web application, supermarket, consumer empowerment, inflation, Kenya, Flask, database management, cost optimization

ACRONYMS & DEFINITION OF TERMS

API - Application Programming Interface: A set of protocols and tools for building software applications that specify how software components should interact.

BeiRadar - The name of the price comparison system (Bei is Swahili for "price").

CSS - Cascading Style Sheets: A stylesheet language used for describing the presentation of HTML documents.

DFD - Data Flow Diagram: A graphical representation of data flow through an information system.

ER Diagram - Entity Relationship Diagram: A visual representation of entity relationships within a database.

Flask - A lightweight Python web framework used for building web applications.

HTML - HyperText Markup Language: The standard markup language for creating web pages.

JavaScript - A programming language that enables interactive web pages.

KNBS - Kenya National Bureau of Statistics: The principal government agency responsible for collecting and disseminating official statistics.

KSh - Kenya Shilling: The official currency of Kenya.

MySQL - An open-source relational database management system.

Python - A high-level programming language used for general-purpose programming.

SQLite - A self-contained, serverless, zero-configuration SQL database engine.

UI - User Interface: The means by which users interact with a computer system.

UX - User Experience: The overall experience of a person using a product or system.

Web Scraping - The automated process of extracting data from websites.

Table of Contents

CHAPTER ONE

1.1 Introduction

The cost of living in Kenya has risen steadily over the past decade, becoming a major challenge for many households. Inflation has affected several sectors, but food prices remain the most significant driver of increased household expenditure. According to Reuters (2025), Kenya's overall inflation rose to 4.5% in August 2025, up from 4.1% the previous month, largely driven by food and transport costs. The Kenya National Bureau of Statistics (KNBS) further reported that the food and non-alcoholic beverages category—which makes up nearly one-third of household spending—rose by 8.3% year-on-year, adding 2.7 percentage points to the overall inflation rate. These statistics show how rising food prices reduce purchasing power and make budgeting increasingly difficult for families.

Supermarkets such as Naivas, Carrefour, QuickMart, and Magunas have become the preferred shopping destinations for many urban consumers due to convenience, variety, and accessibility. However, their prices differ significantly, and the lack of a centralized platform to compare these prices forces consumers to manually check multiple websites or visit stores physically. This process is time-consuming, inefficient, and often leads to overspending or missed promotional opportunities.

Globally, price comparison tools like PriceSpy help shoppers compare products across different retailers. However, these platforms are not localized to Kenya and do not reflect the unique challenges of the Kenyan market. As Kenya's digital sector continues to grow, there is a clear opportunity to develop a localized price comparison system tailored to the needs of Kenyan consumers. This project introduces **BeiRadar**, a web-based supermarket price comparison system designed to help shoppers identify affordable options, improve budgeting, and increase price transparency in an economy affected by rising inflation.

1.2 Background

Kenya's supermarket retail sector has expanded significantly over the years, with leading supermarkets such as Naivas, Carrefour, QuickMart, and Magunas offering a wide variety of household and food products. Although many of these supermarkets have digital platforms—ranging from websites and mobile apps to social media pages—their pricing information remains fragmented. Consumers must browse multiple sources to confirm product prices, especially for essential goods like cooking oil, sugar, rice, and milk.

The rise in food inflation has further complicated the situation. As KNBS and Reuters (2025) highlight, food remains the primary driver of inflation, greatly affecting households that spend a large portion of their income on basic necessities. The inconsistency in supermarket pricing creates a gap where consumers cannot easily determine which store offers better value for specific products.

While international solutions like PriceSpy exist, they do not meet Kenya's local needs or supermarket structure. Local platforms such as Money254 occasionally publish weekly supermarket comparisons, but these are static, not interactive, and do not cover the full range of products consumers need. There is a clear lack of a centralized, real-time price comparison tool that reflects the Kenyan shopping environment.

This background demonstrates the need for a localized system such as BeiRadar, which aims to consolidate supermarket prices, simplify the shopping experience, and empower consumers to make informed choices in a challenging economic environment.

1.3 Emerging Issues

Several emerging issues underscore the importance of developing a supermarket price comparison system in Kenya:

i. **Rising Food Inflation:**
   Food inflation continues to be a major contributor to increased living costs. Even small price differences significantly affect household budgets.

ii. **Fragmented Supermarket Data:**
   Supermarkets publish prices across different platforms—websites, apps, and social media—making it difficult for consumers to access consolidated information.

iii. **Increased Digital Adoption:**
   More consumers are shopping online, but the lack of centralized, updated price data limits the effectiveness of digital shopping.

iv. **Time and Effort Needed for Price Checking:**
   Shoppers must manually compare prices across multiple supermarkets, which is slow, tedious, and impractical.

v. **Lack of Price Transparency:**
   Without clear visibility into price variations and promotions, consumers risk overspending or missing out on more affordable options.

These issues highlight the demand for a solution like BeiRadar, which aims to bring transparency, efficiency, and convenience to the supermarket shopping experience in Kenya.

1.4 Problem Statement

The rising cost of living in Kenya has risen steadily over the years, causing a strain in many family's finances, with food inflation being a number one contributor. Despite the fact that many shops and supermarkets have been deployed online, there is no way to compare the prices and make a cheap and affordable choice of products. As a result, customers are at a risk of being overpriced for products like flour, sugar and milk. With the economy currently it is not strange that people will try to save as much coins as they could.

There are some international platforms like PriceSpy albeit they do not address Kenya's specific needs. Kenya's supermarket publishes their prices in websites and social media but there is no central platform where these prices are aggregated to be used by consumers so they could compare

the prices. Therefore, consumers use more time in searching the best prices or even settling for the nearest shops prices. This creates inefficiency and worsens financial challenges.

The absence of a localized price comparison tool is a gap that continues to disadvantage Kenyan consumers. With no access to reliable, consolidated supermarket price information, family household cannot budget efficiently or make enlightened shopping decisions. This project seeks to address that gap by creating a simple web-based system that provides transparent, up-to-date comparisons of essential supermarket goods in Nairobi.

1.5 Main Objective

To develop a supermarket price comparison web application (BeiRadar) that compiles product prices from major Nairobi supermarkets, enabling consumers to make educated, cost-effective decisions and improve household budgeting in a time of rising inflation.

1.6 Specific Objectives (SMART)

i. To develop a user-friendly web application that consolidates and displays supermarket prices for selected essential goods.

ii. To integrate a product search and filtering system that allows consumers to quickly identify the most affordable options.

iii. To streamline budgeting by implementing a shopping cart feature that calculates total costs across different supermarkets.

iv. To implement a secure and efficient database system that stores and manages supermarket data, product details, and pricing information for easy retrieval and updates.

1.7 Justification

This project is significant because it directly responds to one of Kenya's most pressing socio-economic issues: the rising cost of living. Food and household essentials consume nearly a third of household income, and even minor differences in prices across supermarkets can significantly affect monthly budgets. A system that provides price transparency will allow consumers to plan their spending better, avoid overpaying, and take advantage of discounts or promotions.

Beyond helping individuals, the project has broader importance. It introduces accountability and competition in the retail sector, encouraging supermarkets to adjust their pricing strategies to remain attractive. It also contributes to Kenya's digital transformation agenda by offering a localized solution inspired by global best practices but tailored to the Kenyan market. By empowering shoppers with knowledge, BeiRadar promotes financial literacy, consumer rights, and more efficient retail experiences.

CHAPTER TWO

LITERATURE REVIEW

2.0 Introduction

This chapter reviews existing price-comparison systems and related work at three levels: global, Africa, and Kenya. The aim is to identify what similar systems do, how they operate, and where BeiRadar will add unique value for Kenyan supermarket shoppers.

2.1 Global view

Price-comparison platforms are established and widely used in many countries. A service such as PriceSpy aggregate product listings and offers from many retailers to allow consumers to compare prices, view price histories, and receive alerts for price drops. These platforms demonstrate that centralized comparison engines can reduce search costs for consumers and increase market price transparency.

Key features commonly offered by global platforms include:

- Aggregation of retailer offers for the same product.
- Price-history charts and alerts to help users decide *when* to buy.
- Filtering and sorting by retailer, price, rating, or delivery options. These features inform the basic functional expectations for any price-comparison system and provide a tested design pattern that BeiRadar can adapt to the Kenyan retail context.

2.2 Africa view

In Africa there are price-comparison players, though their focus and maturity vary by country and sector. South Africa's Troli is a notable example: it aggregates offers across many online retailers and includes groceries among other product categories. Similarly, country-specific services and marketplace aggregators exist that compare electronics, appliances, and general retail items. These

services show that comparative shopping tools can be viable in African markets, though adoption for supermarket essentials is less widespread than for electronics or general retail.

Regional learnings for BeiRadar:

- African comparison platforms tend to focus first on high-value / high-margin categories (electronics, appliances) where consumers seek the best deals; groceries are lower margin and present higher data-collection challenges.
- Successful African platforms often combine web data with partnerships and affiliate feeds to ensure data quality and coverage. These lessons suggest BeiRadar will likely need hybrid data collection (scraping + manual/admin updates + partnerships) for supermarket coverage.

2.3 Kenyan view — local landscape and gaps

In Kenya, the retail sector is increasingly digitized: major supermarket chains (Naivas, Carrefour, QuickMart) maintain online shops or mobile apps with product listings and promotions. However, data is fragmented across retailer platforms, promotional social media posts, and occasional weekly price trackers published by local outlets (e.g., Money254's weekly shopping tracker). Consumers who want to compare even a single item (for example, cotton wool or cooking oil) often must check multiple retailer websites or social channels manually.

Existing Kenyan coverage tends to be:

- **Fragmented and manual:** supermarket websites exist but are separate; independent blogs or news sites publish snapshots (weekly trackers) rather than interactive tools for ad-hoc price lookups.
- **Promotion-driven:** many local retail price signals appear first as promotions on social media or flyers (difficult to harvest automatically without OCR or partnerships).

These gaps indicate a clear local need for a consolidated application tailored to Kenya's retail environment: BeiRadar's focus on Nairobi supermarkets, combined with clear source labeling

(e.g., “price from Carrefour site” vs “promo from Instagram”), would address both practical user needs and methodological transparency.

2.4 Unique contribution of BeiRadar

Based on the reviewed literature and market evidence, BeiRadar’s unique contributions will be:

1. **Localized supermarket focus:** Unlike global sites that prioritize wide product categories or international retailers, BeiRadar will focus specifically on essential supermarket goods in Kenya (starting with Nairobi supermarkets), making the product set, units, and promotions directly relevant to Kenyan households. This local specialization helps with unit standardization (e.g., 2kg maize flour) and culturally relevant categories.
2. **Hybrid data strategy:** Learning from African platforms’ need for partnerships and from local fragmentation, BeiRadar will combine automated scraping (where possible) with manual/admin entry and optional retail partnership feeds for coverage and accuracy. This pragmatic hybrid approach is well suited to the Kenyan environment where some retailers publish rich online catalogs while others rely on social media and in-store promotions.
3. **Cart-based comparison & budgeting:** While many platforms compare individual product offers, BeiRadar will prioritize a shopping-cart view that computes total basket cost across supermarkets — a practical feature for budget-conscious consumers (students, families) and particularly useful during periods of food inflation. This functionality responds to the real user pain that motivated the project (time spent checking multiple sites for a single shopping basket).
4. **Transparency & source labeling:** Given the variety of data sources (official online catalogs, social-media promos, manual checks), BeiRadar will explicitly label each price entry by source and timestamp. This transparency helps users weigh the reliability of each price and defend the system’s integrity when defending methodology to evaluators or users.

2.5 Summary and implications for design

The literature and market scan show that price-comparison platforms are effective tools for consumer empowerment globally, that African markets have emerging price-comparison services (though groceries are less covered), and that in Kenya reputable supermarket data exists but is scattered. Therefore, a locally focused, hybrid-data supermarket price comparison system that emphasizes cart-level budgeting and clear source labeling (BeiRadar) is both novel in the Kenyan context and grounded in successful global patterns. These conclusions inform the methodology and design choices in subsequent chapters (scraping strategy, database schema, UI priorities, and data-validation workflows).

CHAPTER THREE

METHODOLOGY

3.0 Introduction

This chapter outlines the methodology used in developing the BeiRadar supermarket price comparison system. It describes the research approach, resources, requirements analysis, design process, and tools that will guide the project. The methodology ensures that the system is user-centered, functional, and adaptable to Kenya's supermarket retail environment.

3.1 Preliminary Investigation

User Requirements Analysis

To understand the needs of potential users, preliminary investigation will involve:

- **Informal Interviews & Surveys**: Targeting university students, low-income households, and regular supermarket shoppers in Nairobi to identify their shopping challenges and preferences.
- **Observation**: Studying how consumers currently access supermarket price information, whether through websites, social media, or physical visits.
- **Document Review**: Checking weekly supermarket trackers (e.g., Money254) and online supermarket platforms (Naivas, Carrefour, QuickMart, Magunas) for existing price data.

User Requirements

The key requirements gathered will include:

- The need to search for specific products and compare prices quickly.
- The ability to calculate the total shopping cost across different supermarkets.
- Transparent updates of discounts and promotions.
- A simple and accessible interface for non-technical users.

Functional Requirements

1. The system shall allow users to search and filter products by name, category, or supermarket.
2. The system shall display comparative prices of selected products across multiple supermarkets.
3. The system shall provide a shopping cart feature to calculate total costs across supermarkets.
4. The system shall allow administrators to update product prices via scraping or manual entry.
5. The system shall display promotions, discounts, or "on-sale" items when available.

Non-Functional Requirements

1. **Usability**: The interface should be simple and intuitive for everyday users.
2. **Performance**: The system should handle queries and return results within 2 seconds.
3. **Reliability**: Price data should be regularly updated (weekly or more frequently).
4. **Scalability**: The system should support expansion to cover more supermarkets and cities in future.
5. **Security**: User queries and data should be handled securely to prevent unauthorized access.

3.2 Design Phase

The system will follow a **three-tier architecture**:

1. **Presentation Layer (Frontend)** – Built using HTML, CSS, and JavaScript to provide a clean, user-friendly interface.
2. **Application Layer (Backend)** – Implemented in Python (Flask or Django) to process requests, connect to the database, and manage product data.

3. **Data Layer (Database)** – A relational database (MySQL or SQLite) to store supermarket names, product details, and prices.

Proposed Architecture

User → Web Interface (Frontend) → Backend API (Python) → Database (MySQL/SQLite) → Output (Price Comparison Table & Cart)

Data Collection Method

- **Web Scraping**: Using Python libraries such as BeautifulSoup and Requests to extract prices from supermarkets that publish online catalogs.
- **Manual Entry**: Where online data is unavailable, data will be manually entered and validated.
- **Hybrid Updates**: Weekly automated scraping combined with admin oversight to ensure accuracy.

System Design Tools

- **Wireframes & Mockups**: For planning user interface layouts.
- **Data Flow Diagram (DFD)**: To show the flow of information between users, backend, and database.
- **Entity Relationship Diagram (ERD)**: To model relationships between entities such as Products, Supermarkets, Prices, and Cart.

3.3 Summary

The methodology emphasizes a user-centered design backed by hybrid data collection. By combining technical design (scraping, database management) with practical requirements analysis (user needs, shopping habits), BeiRadar will address the real gap in Kenya's supermarket retail space. The next chapter (Chapter Four) will present the system analysis, UML designs, and implementation timelines.

CHAPTER FOUR

IMPLEMENTATION

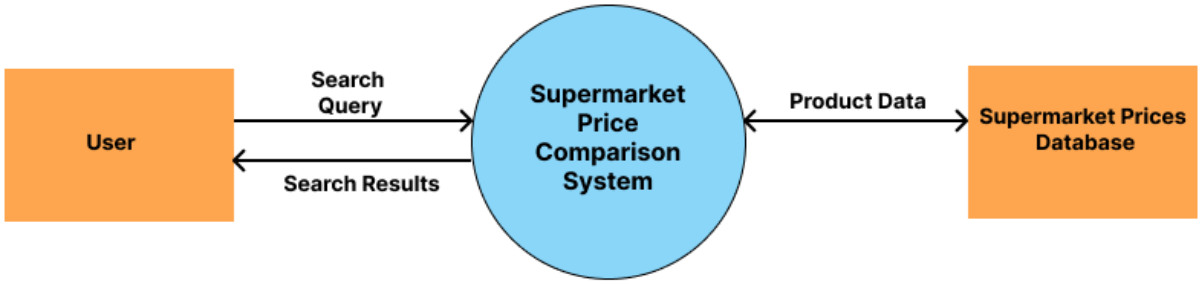
4.0 System Analysis & Design

This chapter presents the detailed system analysis and design for the **BeiRadar** supermarket price comparison web application. The objective is to translate the user requirements and methodology into visual and practical representations of the system, showing how data flows, how entities interact, and how the project schedule is structured. The analysis and design provide a foundation for implementation and ensure clarity for evaluators.

4.1 Data Flow Diagram (DFD)

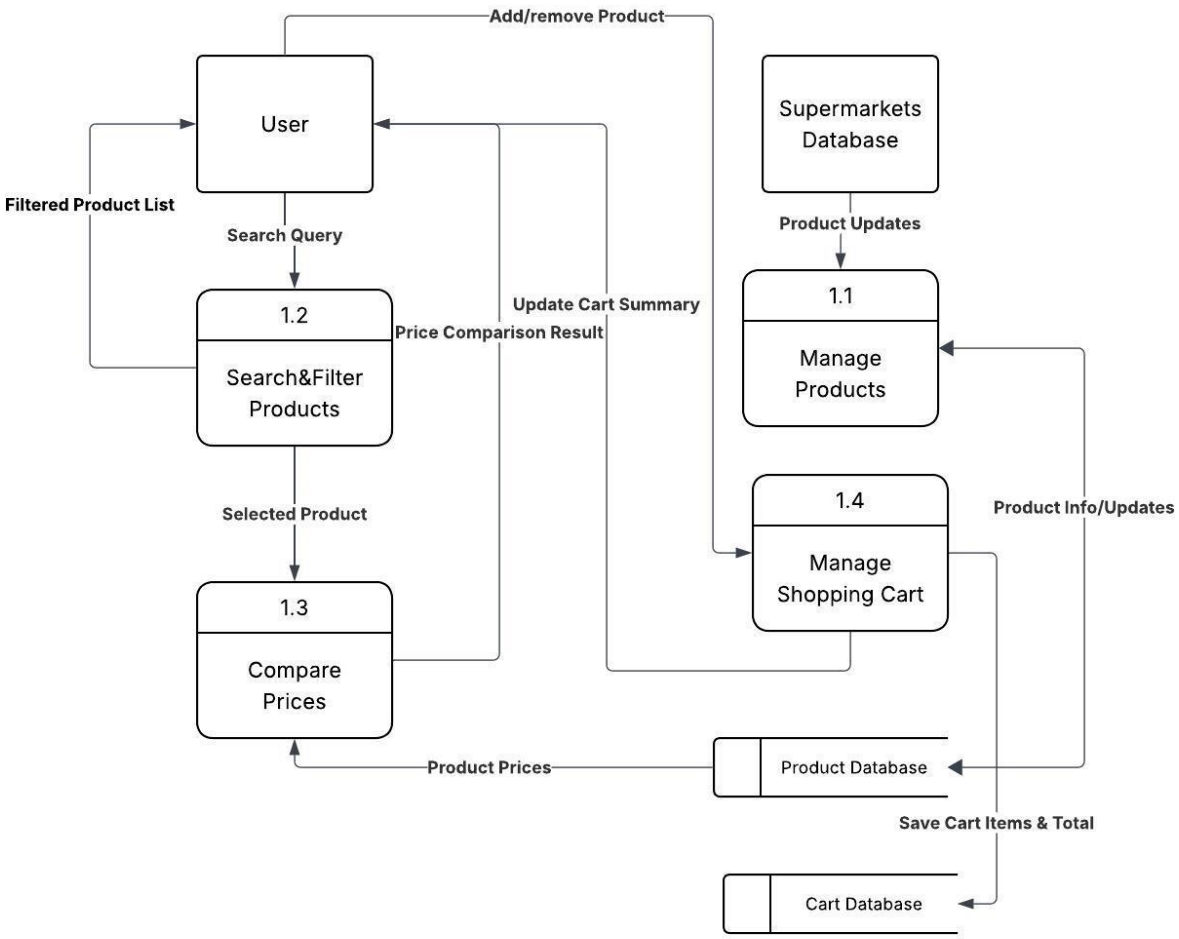
Level 0 DFD

The Level 0 diagram provides a high-level overview of the entire system. It depicts the main processes, external entities, and major data flows without going into detailed internal workings. In this project, Level 0 was created using Figma, which allowed for clear visualization of the system's top-level structure, showing how users interact with the platform and how data moves between the system and external sources like supermarkets and databases.



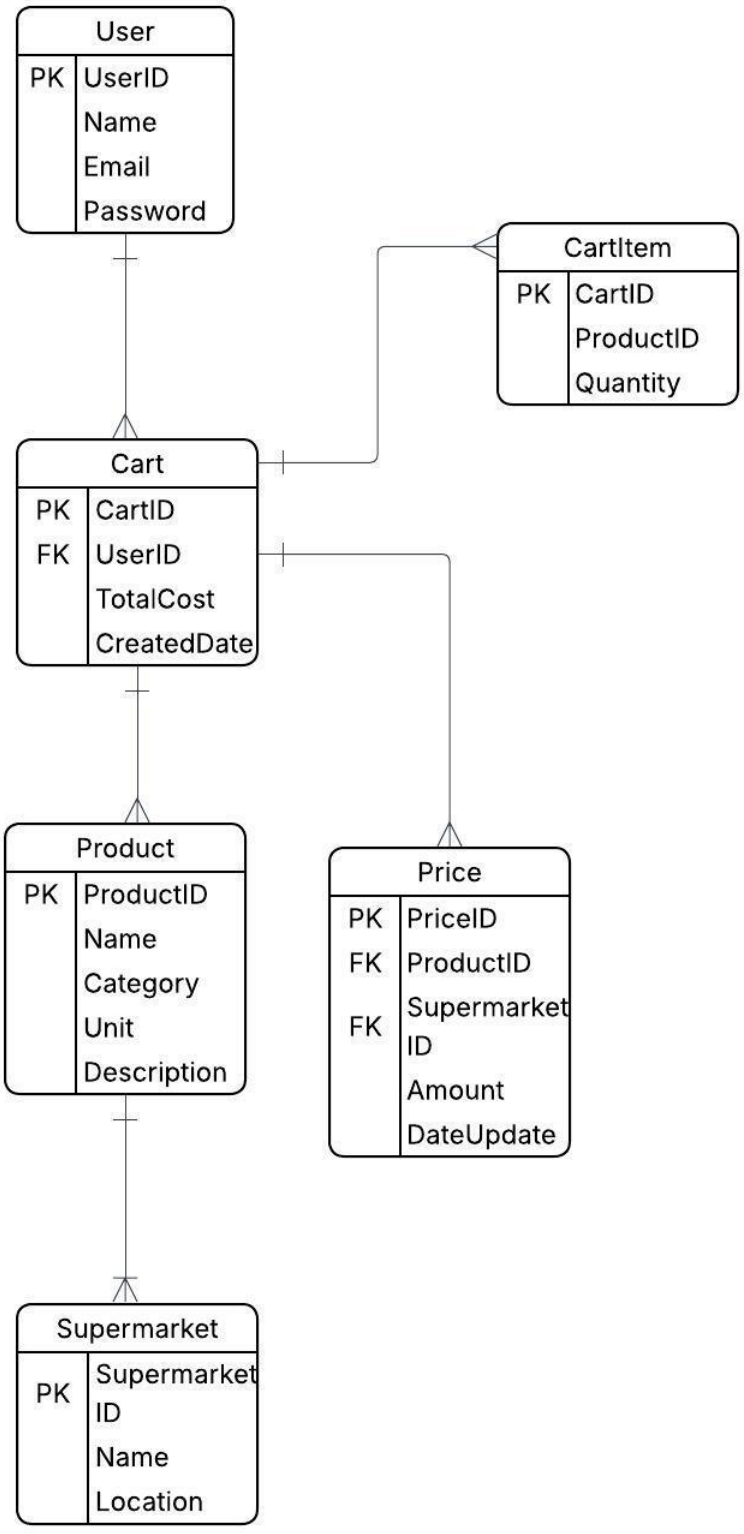
Level 1 DFD

The Level 1 diagram expands each main process from Level 0 into more detailed subprocesses, demonstrating how specific functionalities are handled, including searching products, filtering, comparing prices, and managing the shopping cart. This diagram was developed using Lucidchart, enabling precise representation of data flows, processes, and data stores. Level 1 DFD ensures that every functional requirement of the system is captured in a structured, visual format.



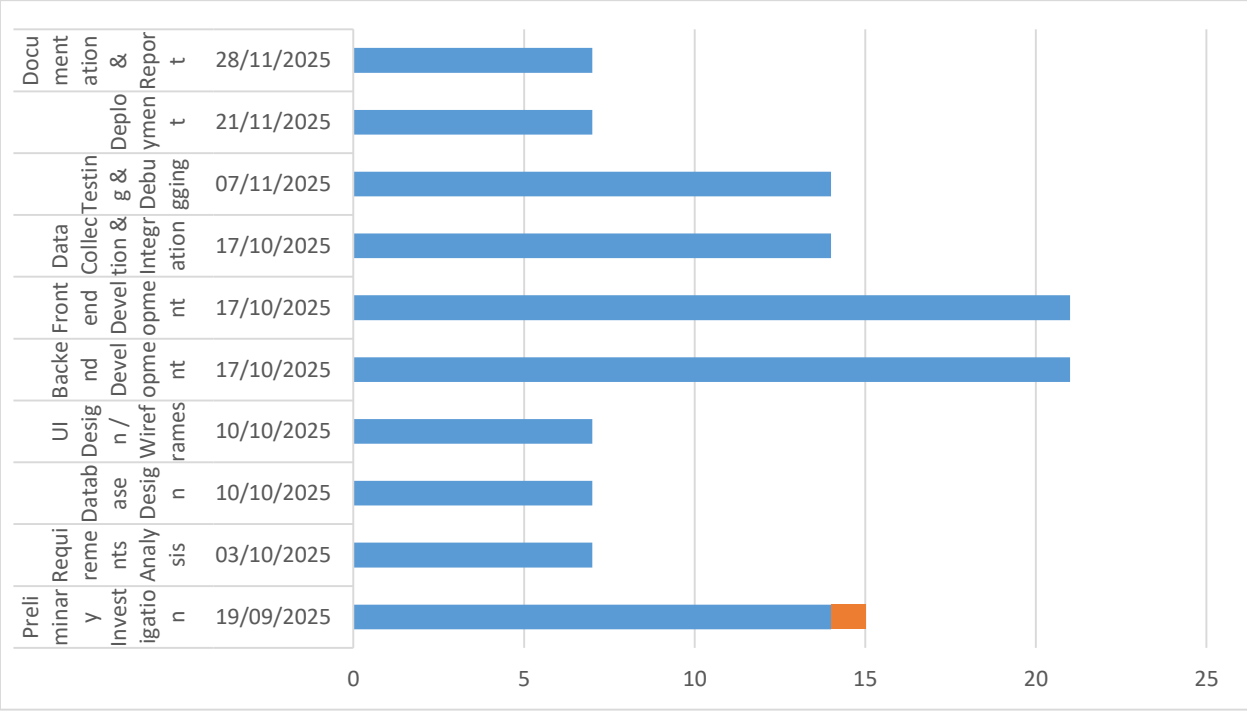
4.2 Entity Relationship (ER) Diagram

The ERD models the relationship between the system's key entities, such as Users, Products, Supermarkets, Prices, and the Shopping Cart. This diagram was also created using Lucidchart, which allowed the use of proper notation for entities, relationships, and cardinality (e.g., one-to-many, many-to-many). The ERD ensures that the database design is normalized and that data is organized efficiently for storage and retrieval.


User
PK UserID
Name
Email
Password
CartItem
PK CartID
ProductID
Quantity
Cart
PK CartID
FK UserID
TotalCost
CreatedDate
Product
PK ProductID
Name
Category
Unit
Description
Price
PK PriceID
FK ProductID
FK Supermarket ID
Amount
DateUpdate
Supermarket
PK Supermarket ID
Name
Location

4.3 Proposed Timelines

A Gantt Chart was developed to illustrate the project schedule and task dependencies. The chart outlines all major tasks, including Preliminary Investigation, Requirements Analysis, Database Design, UI Design, Backend Development, Frontend Development, Data Collection & Integration, Testing, Deployment, and Documentation. For this project, the Gantt Chart was created using Microsoft Excel, allowing automatic calculation of task durations, start and end dates, and visualization of overlapping and sequential tasks. This timeline ensures that the project is planned efficiently and deadlines are manageable.



CHAPTER FIVE

CODING

5.1 System Code

1. **Backend Code(Flask)**

   webapp.py

```
from flask import Flask, render_template, request, session, redirect, url_for
from pyngrok import ngrok, conf
import sqlite3
import os
from dotenv import load_dotenv

# Load environment variables
load_dotenv()

app = Flask(__name__)
# Use environment variable, fallback to secure default
app.secret_key = os.getenv('SECRET_KEY', 'dev-key-change-in-production')

DB_PATH = 'beiradar.db'
PORT = 5000

# NGROK CONFIG

NGROK_PATH = r"C:\Users\Lilian Imma W\Downloads\ngrok-v3-stable-windows-amd64\ngrok.exe"

conf.get_default().ngrok_path = NGROK_PATH
ngrok.kill()

```

```
public_url = ngrok.connect(PORT)
print(f"Shareable Ngrok URL: {public_url}")

# DATABASE FUNCTIONS

def get_products(search_query=None, category=None, min_price=None, max_price=None, min_discount=None):
"""
Fetch products from the database with optional filtering.
Searches by BOTH product name AND category.
"""
conn = sqlite3.connect(DB_PATH)
conn.row_factory = sqlite3.Row
cursor = conn.cursor()

sql = "SELECT rowid AS id, * FROM products"
conditions = []
params = []

if search_query:
# Search by product name OR category
search_lower = f"%{search_query.lower()}%"
conditions.append("(LOWER(product) LIKE ? OR LOWER(category) LIKE ?)")
params.append(search_lower)
params.append(search_lower)

if category:
conditions.append("LOWER(category) = ?")
params.append(category.lower())

if conditions:
sql += " WHERE " + " AND ".join(conditions)

cursor.execute(sql, params)
```

```
rows = cursor.fetchall()
conn.close()

products = [dict(row) for row in rows]

# Apply price and discount filtering (post-database)
if min_price is not None or max_price is not None or min_discount is not None:
filtered = []
for p in products:
best_price, _ = calculate_best_price(p)

# Check price range
if best_price:
if min_price is not None and best_price < min_price:
continue
if max_price is not None and best_price > max_price:
continue

# Check discount
if min_discount is not None:
stores = ['Carrefour', 'Naivas', 'Quickmart']
max_product_discount = 0
for store in stores:
store_lower = store.lower()
current = p.get(f'{store_lower}_current')
original = p.get(f'{store_lower}_original')

if current and original and original != '-':
try:
curr_f = float(current)
orig_f = float(original)
if orig_f > curr_f:
discount = (orig_f - curr_f) * 100 / orig_f
```

```
    max_product_discount = max(max_product_discount, discount)
    except:
    pass

    if max_product_discount < min_discount:
    continue

    filtered.append(p)
    products = filtered

    return products
    def get_categories():
    conn = sqlite3.connect(DB_PATH)
    cursor = conn.cursor()
    cursor.execute("SELECT DISTINCT category FROM products ORDER BY category")
    categories = [row[0] for row in cursor.fetchall()]
    conn.close()
    return categories

    @app.context_processor
    def utility_processor():
    return dict(get_categories=get_categories)

    def get_cart():
    """Return the cart dict from session."""
    return session.get('cart', {})

    def save_cart(cart):
    """Save the cart dict into session."""
    session['cart'] = cart

    def add_to_cart(product_name, quantity=1):
    cart = get_cart()
    pname = str(product_name)
```

```
    if pname in cart:
    cart[pname] += quantity
    else:
    cart[pname] = quantity
    save_cart(cart)

    def remove_from_cart(product_name):
    cart = get_cart()
    pname = str(product_name)
    if pname in cart:
    del cart[pname]
    save_cart(cart)

    def update_cart(product_name, quantity):
    cart = get_cart()
    pname = str(product_name)
    if quantity <= 0:
    cart.pop(pname, None)
    else:
    cart[pname] = quantity
    save_cart(cart)

    # PRICE CALCULATION FUNCTIONS

    def calculate_best_price(product):
    """
    Calculate the best price from all stores.
    Returns None for best_store if all prices are the same.
    """
    stores = [
    ('Carrefour', product.get('carrefour_current')),
    ('Naivas', product.get('naivas_current')),
    ('Quickmart', product.get('quickmart_current'))
    ]
```

```
    # Filter out invalid prices
    valid_prices = [(store, price) for store, price in stores if price not in (None, 0)]

    if not valid_prices:
        return None, None

    # Find the best (minimum) price
    best_store, best_price = min(valid_prices, key=lambda x: x[1])

    # Check if all valid prices are the same
    unique_prices = set(price for _, price in valid_prices)

    # If all prices are the same, don't highlight any store as "best"
    if len(unique_prices) == 1:
        return best_price, None  # Return price but no best_store

    return best_price, best_store

def get_image_url(filename):
    """Returns the correct URL for a product image."""
    if not filename:
        return "https://via.placeholder.com/200?text=No+Image"

    filename = filename.replace("images/", "").strip()
    return url_for('static', filename=f'images/{filename}')

app.jinja_env.globals.update(get_image_url=get_image_url)

def process_products(products):
    """Process raw product data into display-friendly format"""
    processed = []
```

```
for p in products:
best_price, best_store = calculate_best_price(p)

stores_data = {}
for store_name in ['Carrefour', 'Naivas', 'Quickmart']:
store_lower = store_name.lower()
current = p.get(f'{store_lower}_current')
original_raw = p.get(f'{store_lower}_original')

if current is not None:
try:
current = float(current)
except:
current = None

original = None
if original_raw and original_raw != '-':
try:
original = float(original_raw)
except:
original = None

discount = 0
if current and original and original > current:
discount = round((original - current) * 100 / original, 2)

stores_data[store_name] = {
'current': current,
'original': original,
'discount': discount,
'display': f"KSh {current:,.0f}" if current else 'N/A'
}

processed.append({
```

```
    'id': p.get('id'),
    'name': p.get('product', 'Unknown Product'),
    'weight': p.get('weight', ''),
    'category': p.get('category', '').replace('_', ' ').title(),
    'best_price': best_price,
    'best_store': best_store,
    'stores': stores_data,
    'typical_price': p.get('cheapest_price'),
    'on_sale': any(d['discount'] > 0 for d in stores_data.values()),
    'image_url': p.get('image_url', ''),
    'is_discounted': bool(p.get('is_discounted_anywhere', 0))
    })

    return processed

    # CART COMPARISON FUNCTIONS

    def calculate_cart_totals_by_store(cart_items):
    """Calculate total cost for the cart at each supermarket."""
    stores = ['Carrefour', 'Naivas', 'Quickmart']
    store_totals = {store: 0 for store in stores}

    for item in cart_items:
    product = item['product_data']
    quantity = item['quantity']

    for store in stores:
    store_lower = store.lower()
    price = product.get(f'{store_lower}_current')
    if price:
    try:
    store_totals[store] += float(price) * quantity
    except (ValueError, TypeError):
    pass
```

```
•
•   valid_totals = {k: v for k, v in store_totals.items() if v > 0}
•
•   if valid_totals:
•   best_store = min(valid_totals, key=valid_totals.get)
•   best_price = valid_totals[best_store]
•   max_savings = max(valid_totals.values()) - best_price
•   else:
•   best_store = None
•   best_price = 0
•   max_savings = 0
•
•   return {
•   'by_store': store_totals,
•   'best_store': best_store,
•   'best_price': best_price,
•   'max_savings': max_savings,
•   'has_all_stores': all(v > 0 for v in store_totals.values())
•   }
•
•   # ROUTES
•
•   @app.route('/', methods=['GET'])
•   def home():
•   """Home route with search and filtering"""
•   query = request.args.get('search', '').strip()
•   min_price = request.args.get('min_price')
•   max_price = request.args.get('max_price')
•   min_discount = request.args.get('min_discount')
•
•   min_price = float(min_price) if min_price else None
•   max_price = float(max_price) if max_price else None
•   min_discount = float(min_discount) if min_discount else None
•
```

```
if query:
products = get_products(
search_query=query,
min_price=min_price,
max_price=max_price,
min_discount=min_discount
)
results = process_products(products)
else:
results = []

return render_template(
'index.html',
query=query,
products=results,
min_price=min_price,
max_price=max_price,
min_discount=min_discount
)

# Category mapping
CATEGORY_MAPPING = {
"Rice": "Rice",
"Cooking Oil": "Oil",
"Sugar": "Sugar",
"Milk": "Milk",
"Yoghurt": "Yoghurt",
"Cheese": "Cheese",
"Laundry/Detergents": "Laundry/Detergents",
"Dishwashing": "Dishwashing",
"Paper products": "Paper products",
"Toothpaste": "Toothpaste",
"Lotion": "Lotion",
"Sanitary Pads": "Sanitary Pads"
```

```
}

CATEGORIES = {
"Foodstuff": ["Rice", "Cooking Oil", "Sugar"],
"Dairy": ["Milk", "Yoghurt", "Cheese"],
"Household": ["Laundry", "Dishwashing", "Paper products"],
"Personal Care": ["Toothpastes", "Lotion", "Sanitary Pads"]
}

@app.route('/api/search-suggestions')
def search_suggestions():
"""API endpoint for autocomplete - Returns products AND categories"""
query = request.args.get('q', '').strip().lower()

if not query or len(query) < 2:
return {'suggestions': []}

suggestions = []
seen = set()

# Get matching products
products = get_products(search_query=query)

for p in products[:10]:  # Limit to 10 products
product_name = p.get('product', '').strip()
if product_name and product_name.lower() not in seen:
suggestions.append({
'text': product_name,
'type': 'product'
})
seen.add(product_name.lower())

# Get matching categories
conn = sqlite3.connect(DB_PATH)
```

```
cursor = conn.cursor()
cursor.execute(
"SELECT DISTINCT category FROM products WHERE LOWER(category) LIKE ? ORDER BY category",
(f"%{query}%",)
)
categories = [row[0] for row in cursor.fetchall()]
conn.close()

for category in categories:
category_display = category.replace('_', ' ').title()
if category_display.lower() not in seen:
suggestions.append({
'text': category_display,
'type': 'category'
})
seen.add(category_display.lower())

# Return formatted suggestions (limit to 12 total)
return {
'suggestions': suggestions[:12]
}
@app.route("/categories")
def categories_list():
categories = [{"name": name, "slug": name.lower().replace(" ", "-")} for name in CATEGORIES.keys()]
return render_template("categories.html", categories=categories)

@app.route("/categories/<category_slug>")
def category_detail(category_slug):
category_name = None
for name in CATEGORIES.keys():
if name.lower().replace(" ", "-") == category_slug:
category_name = name
```

```
    break

    if not category_name:
    return "Category not found", 404

    subcategories = CATEGORIES[category_name]
    return render_template("category_detail.html", category_name=category_name,
    subcategories=subcategories)

    @app.route('/products/<subcategory_slug>')
    def products_by_subcategory(subcategory_slug):
    """Show products in subcategory with filtering"""
    subcategory_name = subcategory_slug.replace('-', ' ').title()
    db_category = CATEGORY_MAPPING.get(subcategory_name, subcategory_name)

    min_price = request.args.get('min_price')
    max_price = request.args.get('max_price')
    min_discount = request.args.get('min_discount')

    min_price = float(min_price) if min_price else None
    max_price = float(max_price) if max_price else None
    min_discount = float(min_discount) if min_discount else None

    products = get_products(
    category=db_category,
    min_price=min_price,
    max_price=max_price,
    min_discount=min_discount
    )
    results = process_products(products)

    return render_template(
    'category_products.html',
    category_name=subcategory_name,
```

```
    products=results,
    min_price=min_price,
    max_price=max_price,
    min_discount=min_discount
    )

    # CART ROUTES

    @app.route("/cart")
    def cart_view():
    """Cart view with multi-store totals"""
    cart = get_cart()
    products_in_cart = []

    for product_name, qty in cart.items():
    products = get_products(search_query=product_name)
    product = products[0] if products else None

    if product:
    processed = process_products([product])[0]
    products_in_cart.append({
    'name': processed['name'],
    'quantity': qty,
    'product_data': product,
    'best_price': processed['best_price'],
    'best_store': processed['best_store'],
    'stores': processed['stores'],
    'category': processed['category'],
    'image_url': processed['image_url']
    })

    cart_summary = calculate_cart_totals_by_store(products_in_cart) if
    products_in_cart else {
    'by_store': {'Carrefour': 0, 'Naivas': 0, 'Quickmart': 0},
```

```
    'best_store': None,
    'best_price': 0,
    'max_savings': 0,
    'has_all_stores': False
    }

    return render_template(
    "cart.html",
    products=products_in_cart,
    cart_summary=cart_summary
    )

    @app.route("/cart/add/<product_name>", methods=['POST'])
    def cart_add(product_name):
    """Add item to cart"""
    qty = int(request.form.get('quantity', 1))
    add_to_cart(product_name, qty)
    return redirect(request.referrer or url_for('home'))

    @app.route("/cart/remove/<product_name>")
    def cart_remove(product_name):
    """Remove item from cart"""
    remove_from_cart(product_name)
    return redirect(url_for('cart_view'))

    @app.route("/cart/update/<product_name>", methods=['POST'])
    def cart_update(product_name):
    """Update item quantity"""
    qty = int(request.form.get('quantity', 1))
    update_cart(product_name, qty)
    return redirect(url_for('cart_view'))

    # OTHER ROUTES

```

```
@app.route('/about')
def about():
return render_template('about.html')

@app.route('/deals')
def deals():
"""Show best deals"""
products = get_products()
deals_list = []

for p in products:
prices = {
'Carrefour': p.get('carrefour_current'),
'Naivas': p.get('naivas_current'),
'Quickmart': p.get('quickmart_current')
}

valid_prices = {k: v for k, v in prices.items() if v and v > 0}

if len(valid_prices) >= 2:
best_store = min(valid_prices, key=valid_prices.get)
best_price = valid_prices[best_store]
max_price = max(valid_prices.values())

if max_price > best_price:
discount_pct = round((max_price - best_price) * 100 / max_price, 2)

if discount_pct > 0:
deals_list.append({
'product_name': p.get('product', 'Unknown'),
'weight': p.get('weight', ''),
'store': best_store,
'old_price': max_price,
'new_price': best_price,
```

```
    'deal_percentage': discount_pct,
    'category': p.get('category', '').replace('_', ' ').title(),
    'image_url': p.get('image_url', '')
    })

    deals_list = sorted(deals_list, key=lambda x: x['deal_percentage'],
    reverse=True)
    return render_template('deals.html', deals=deals_list)

    # CUSTOM FILTERS

    @app.template_filter('safe_price')
    def safe_price(value):
    """
    Safely format a price value, handling None and invalid values.
    Returns 'N/A' if value is None or invalid.
    """
    if value is None:
    return "N/A"
    try:
    return f"KSh {float(value):,.0f}"
    except (ValueError, TypeError):
    return "N/A"

    # ERROR HANDLERS

    @app.errorhandler(404)
    def page_not_found(e):
    return render_template('404.html'), 404

    @app.errorhandler(500)
    def internal_error(e):
    return render_template('500.html'), 500

```

```
# RUN APP

if __name__ == '__main__':
print(f"Visit your app locally: http://127.0.0.1:{PORT}")
print(f"Shareable Ngrok URL: {public_url}")
app.run(host="0.0.0.0", port=PORT, debug=True, use_reloader=False)
```

2. **Database Seeder**

   Seeder_db.py

```
import pandas as pd
import sqlite3

# Load your Excel files
files = {
"oil": "Sort/Oil data.xlsx",
"rice": "Sort/Rice Data.xlsx",
"sugar": "Sort/Sugar data.xlsx",
"milk": "Sort/Milk data.xlsx",
"yoghurt": "Sort/Yoghurt data.xlsx",
"cheese": "Sort/Cheese data.xlsx",
"laundry": "Sort/Laundry data.xlsx",
"dishwashing": "Sort/Dishwashing data.xlsx",
"paper products": "Sort/Paper data.xlsx",
"sanitary pads": "Sort/Sanitary Pads data.xlsx",
"toothpastes": "Sort/Toothpaste data.xlsx",
"lotion": "Sort/Lotion data.xlsx",


}

dfs = {}
for category, path in files.items():
df = pd.read_excel(path)
```

```
# Normalize column names
df.columns = [c.lower().replace(' ', '_') for c in df.columns]

# Add category column
df['category'] = category

# Optional: add extra columns
df['image_url'] = ''
df['is_discounted_anywhere'] = 1

# Convert price columns to numeric
for col in df.columns:
if 'price' in col:
df[col] = pd.to_numeric(df[col], errors='coerce')

dfs[category] = df

# Concatenate all products
all_products = pd.concat(dfs.values(), ignore_index=True)

# Insert into SQLite database
conn = sqlite3.connect('beiradar.db')
all_products.to_sql('products', conn, if_exists='replace', index=False)
conn.close()

print("Database updated successfully!")
```

3. **Database Image Mapper**

   Mapper.py

```
import sqlite3
import os
import re
```

```
from difflib import SequenceMatcher

DB_PATH = 'beiradar.db'
IMAGE_FOLDER = 'static/images'

# Connect to database
conn = sqlite3.connect(DB_PATH)
cursor = conn.cursor()

# Fetch all products
cursor.execute("SELECT product FROM products")
products = [p[0] for p in cursor.fetchall()]

# List all images in folder
image_files = os.listdir(IMAGE_FOLDER)

# Manual mappings for difficult matches
MANUAL_MAPPINGS = {
'Daawat Long Grain 5kg': 'dawaat lgrain 5kg',
'Daawat Long Grain 1Kg': 'daawat lgrain 2kg',  # Use 2kg image if 1kg doesn't exist
'Sunlight 2 in 1 Hand Washing Powder Lavender Sensations 500g': 'sunlight 2 in 1
Lavender 500g',
'Kleenex Toilet Paper Roll White 8pcs': 'kleenex tpaper white 8pcs',
'Molped Sanitary Pads Ultra Soft 16pcs': 'molped spads 16pcs',
'Dabur Herbal Toothpaste Clove 150g': 'dabur clove toothpaste 150g',
'Nice & Lovely Body Lotion Cocoa Butter 400ml': 'nice and lovely cocoa butter
400ml',
}

# Expanded abbreviations mapping
abbreviations = {
'sberry': 'strawberry',
'nat': 'natural',
'mlk': 'milk',
'pcs': 'pcs',
```

```
'l': 'liter',
'kg': 'kilogram',
'g': 'gram',
'ltr': 'liter',
'tbsp': 'tablespoon',
'tsp': 'teaspoon',
'ml': 'milliliter',
'oz': 'ounce',
'fr': 'fresh',
'veg': 'vegetable',
'choc': 'chocolate',
'straw': 'strawberry',
'van': 'vanilla',
'lgrain': 'long grain',
}

def clean_text(text):
"""Clean and normalize text for comparison"""
text = text.lower()
text = text.replace('-', ' ').replace('–', ' ').replace('_', ' ')
text = text.replace('&', 'and')
text = re.sub(r'[^a-z0-9\s]', '', text)

# Expand abbreviations
tokens = text.split()
expanded = [abbreviations.get(tok, tok) for tok in tokens]
return ' '.join(expanded)

def extract_key_terms(text):
"""Extract key terms (brand, product type, size) from text"""
cleaned = clean_text(text)

# Remove common filler words
stopwords = {'the', 'a', 'an', 'and', 'or', 'of', 'for', 'with', 'in', 'hand',
'washing', 'powder'}
```

```
tokens = [t for t in cleaned.split() if t not in stopwords and len(t) > 1]

return tokens

def similarity_score(str1, str2):
"""Calculate similarity between two strings"""
return SequenceMatcher(None, str1, str2).ratio()

def find_best_image(product_name, debug=False):
"""Find best matching image for a product"""
# Check manual mappings first
if product_name in MANUAL_MAPPINGS:
target_filename = MANUAL_MAPPINGS[product_name]
for img in image_files:
img_name_no_ext = os.path.splitext(img)[0]
if clean_text(img_name_no_ext) == clean_text(target_filename):
return f"images/{img}"
# Also try fuzzy match on manual mapping
if     clean_text(target_filename)     in     clean_text(img_name_no_ext)     or
clean_text(img_name_no_ext) in clean_text(target_filename):
return f"images/{img}"

product_clean = clean_text(product_name)
product_tokens = extract_key_terms(product_name)

best_match = None
best_score = 0

for img in image_files:
img_name = os.path.splitext(img)[0]
img_clean = clean_text(img_name)
img_tokens = extract_key_terms(img_name)

# Strategy 1: Exact match
if product_clean == img_clean:
```

```
return f"images/{img}"

# Strategy 2: All product tokens in image
if all(token in img_clean for token in product_tokens):
return f"images/{img}"

# Strategy 3: Score-based matching
# Count matching tokens
matching_tokens = sum(1 for token in product_tokens if token in img_tokens)
token_ratio = matching_tokens / max(len(product_tokens), 1)

# Calculate string similarity
string_sim = similarity_score(product_clean, img_clean)

# Combined score (weighted)
score = (token_ratio * 0.7) + (string_sim * 0.3)

if score > best_score and score > 0.48:  # Lowered threshold slightly
best_score = score
best_match = img

if debug and score > 0.3:
print(f"  {img_name}: token_ratio={token_ratio:.2f}, string_sim={string_sim:.2f},
score={score:.2f}")

if best_match:
return f"images/{best_match}"

return None

# Update database with progress tracking
unmatched_products = []
matched_count = 0

print("Matching products to images...\n")
```

```
for i, product in enumerate(products, 1):
image_path = find_best_image(product)

if image_path:
cursor.execute("""
UPDATE products
SET image_url = ?
WHERE product = ?
""", (image_path, product))
matched_count += 1
print(f"✓        [{i}/{len(products)}]        Matched:        {product}        →
{os.path.basename(image_path)}")
else:
unmatched_products.append(product)
print(f"X [{i}/{len(products)}] No match: {product}")

conn.commit()
conn.close()

# Summary
print("\n" + "="*70)
print(f"SUMMARY:")
print(f"Total products: {len(products)}")
print(f"Matched: {matched_count}")
print(f"Unmatched: {len(unmatched_products)}")
print("="*70)

# Show unmatched products for manual review
if unmatched_products:
print("\nProducts still needing manual review:")
for p in unmatched_products:
print(f" - {p}")
# Show debug info for unmatched
print(f"   Debug matches:")
```

```
find_best_image(p, debug=True)
print()
else:
print("\n🎉 All products matched successfully!")

# Show available images that might be good candidates
if unmatched_products:
print("\n" + "="*70)
print("Available image files for reference:")
print("="*70)
for img in sorted(image_files):
print(f"  - {img}")
```

4. **Database Inspector Tool**

   Db_checker.py

```
import sqlite3

DB_PATH = 'beiradar.db'

def check_database():
"""Check database structure and content"""
conn = sqlite3.connect(DB_PATH)
cursor = conn.cursor()

print("=" * 60)
print("DATABASE DIAGNOSTIC")
print("=" * 60)

# Check if products table exists
cursor.execute("SELECT name FROM sqlite_master WHERE type='table' AND
name='products'")
table_exists = cursor.fetchone()

if not table_exists:
```

```
print("❌ ERROR: 'products' table does not exist!")
conn.close()
return

print("✓ Table 'products' exists\n")

# Get column names
cursor.execute("PRAGMA table_info(products)")
columns = cursor.fetchall()
print(f"Columns in 'products' table ({len(columns)} total):")
for col in columns:
print(f"  - {col[1]} ({col[2]})")

print()

# Count total products
cursor.execute("SELECT COUNT(*) FROM products")
total = cursor.fetchone()[0]
print(f"Total products: {total}")

if total == 0:
print("❌ WARNING: No products in database!")
conn.close()
return

print()

# Show sample products
cursor.execute("SELECT * FROM products LIMIT 3")
rows = cursor.fetchall()
col_names = [description[0] for description in cursor.description]

print("Sample products:")
for i, row in enumerate(rows, 1):
print(f"\n--- Product {i} ---")
```

```
for col_name, value in zip(col_names, row):
print(f"  {col_name}: {value}")

print()

# Check for products with 'milk' in name
cursor.execute("SELECT COUNT(*) FROM products WHERE LOWER(product) LIKE '%milk%'")
milk_count = cursor.fetchone()[0]
print(f"Products containing 'milk': {milk_count}")

if milk_count > 0:
cursor.execute("SELECT product FROM products WHERE LOWER(product) LIKE '%milk%'
LIMIT 5")
milk_products = cursor.fetchall()
print("Sample milk products:")
for prod in milk_products:
print(f"  - {prod[0]}")

print()

# Check categories
cursor.execute("SELECT DISTINCT category FROM products")
categories = cursor.fetchall()
print(f"Categories ({len(categories)} total):")
for cat in categories:
print(f"  - {cat[0]}")

conn.close()
print("\n" + "=" * 60)

if __name__ == '__main__':
try:
check_database()
except sqlite3.OperationalError as e:
print(f"❌ ERROR: Could not connect to database")
```

```
print(f"   {e}")
print("\nMake sure 'beiradar.db' exists in the current directory.")
except Exception as e:
print(f"❌ ERROR: {e}")
```

5. Database Previewer

Preview_db.py

```
import sqlite3

DB_PATH = 'beiradar.db'

conn = sqlite3.connect(DB_PATH)
cursor = conn.cursor()

# See the first 10 products
cursor.execute("SELECT rowid, product, category, image_url FROM products LIMIT
10")
rows = cursor.fetchall()

for row in rows:
print(f"ID: {row[0]}, Product: {row[1]}, Category: {row[2]}, Image: {row[3]}")

conn.close()
```

Frontend

6. index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar — Price Search</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Filter Section */
.filter-section {
background: white;
border-radius: 16px;
padding: 25px;
margin: 30px auto;
max-width: 1200px;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
}

.filter-title {
font-size: 1.1rem;
font-weight: 700;
color: #111;
margin-bottom: 15px;
}

.filters-container {
display: grid;
grid-template-columns: repeat(auto-fit, minmax(180px, 1fr));
gap: 15px;
}

.filter-group {
display: flex;
flex-direction: column;
gap: 8px;
}

.filter-group label {
font-weight: 600;
```

```
color: #333;
font-size: 0.95rem;
}

.filter-group input {
padding: 10px 12px;
border: 2px solid #ddd;
border-radius: 8px;
font-size: 0.95rem;
transition: border-color 0.2s;
}

.filter-group input:focus {
outline: none;
border-color: #667eea;
}

.filter-actions {
display: flex;
gap: 10px;
grid-column: 1 / -1;
}

.filter-btn {
flex: 1;
padding: 12px 20px;
border: none;
border-radius: 8px;
font-weight: 600;
font-size: 1rem;
cursor: pointer;
transition: transform 0.2s;
}

.filter-btn-apply {
```

```
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
}

.filter-btn-apply:hover {
transform: scale(1.05);
}

.filter-btn-reset {
background: #f0f0f0;
color: #333;
}

.filter-btn-reset:hover {
background: #e0e0e0;
}

.active-filters {
display: flex;
flex-wrap: wrap;
gap: 10px;
margin-top: 15px;
}

.filter-chip {
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 8px 12px;
border-radius: 20px;
font-size: 0.85rem;
display: flex;
align-items: center;
gap: 8px;
}
```

```
.filter-chip button {
background: none;
border: none;
color: white;
cursor: pointer;
font-size: 1.1rem;
padding: 0;
}

/* Product Grid */
.products-grid {
display: grid;
grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
gap: 25px;
padding: 30px;
max-width: 1400px;
margin: 0 auto;
}

.product-card {
background: white;
border-radius: 16px;
padding: 20px;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
transition: transform 0.2s, box-shadow 0.2s;
display: flex;
flex-direction: column;
}

.product-card:hover {
transform: translateY(-8px);
box-shadow: 0 8px 20px rgba(0, 0, 0, 0.12);
}

.product-image-container {
```

```
width: 100%;
height: 200px;
display: flex;
align-items: center;
justify-content: center;
margin-bottom: 15px;
background: #f9f9f9;
border-radius: 12px;
padding: 15px;
}

.product-image {
max-width: 100%;
max-height: 100%;
object-fit: contain;
}

.product-card h3 {
font-size: 1rem;
font-weight: 600;
color: #111;
margin: 0 0 8px 0;
text-align: center;
line-height: 1.4;
min-height: 2.8em;
}

.product-category {
color: #666;
font-size: 0.85rem;
text-align: center;
margin-bottom: 12px;
}

.best-price-banner {
```

```
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 12px 15px;
border-radius: 12px;
margin-bottom: 15px;
text-align: center;
}

.best-price-banner .label {
font-size: 0.75rem;
opacity: 0.9;
}

.best-price-banner .price {
font-size: 1.3rem;
font-weight: 700;
}

.store-prices {
display: flex;
flex-direction: column;
gap: 8px;
margin-bottom: 15px;
}

.store-price-card {
background: #f9f9f9;
border: 2px solid transparent;
border-radius: 10px;
padding: 10px 12px;
display: flex;
justify-content: space-between;
align-items: center;
font-size: 0.9rem;
```

```
transition: all 0.2s;
}

.store-price-card:hover {
background: #fff;
border-color: #667eea;
}

.store-price-card.best-deal {
background: linear-gradient(135deg, #d4fc79 0%, #96e6a1 100%);
border-color: #4caf50;
}

.store-price-card.unavailable {
opacity: 0.6;
}

.store-price-card .store-name {
font-weight: 600;
color: #333;
font-size: 0.9rem;
}

.store-price-card .price {
font-weight: 700;
color: #111;
}

.add-to-cart-btn {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
padding: 12px 15px;
border-radius: 10px;
```

```
font-size: 0.95rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
margin-top: auto;
}

.add-to-cart-btn:hover {
transform: scale(1.05);
}

.quantity-input {
width: 100%;
padding: 8px;
border: 2px solid #ddd;
border-radius: 8px;
text-align: center;
font-size: 0.95rem;
font-weight: 600;
margin-bottom: 10px;
}

.sec-title {
text-align: center;
font-size: 2rem;
font-weight: 700;
color: #111;
margin: 30px 0 20px 0;
}

.no-results {
text-align: center;
padding: 60px 20px;
color: #666;
}
```

```
.no-results p {
font-size: 1.1rem;
}

.welcome-message {
text-align: center;
padding: 60px 20px;
color: #666;
}

.welcome-message h2 {
font-size: 1.8rem;
color: #111;
margin-bottom: 10px;
}

/* Autocomplete Styles */
.search-container {
position: relative;
max-width: 600px;
margin: 0 auto;
}

.search-box {
display: flex;
gap: 10px;
margin-bottom: 20px;
}

.search-input-wrapper {
flex: 1;
position: relative;
}
```

```
#searchInput {
width: 100%;
padding: 15px 20px;
border: 2px solid #ddd;
border-radius: 12px;
font-size: 1rem;
transition: border-color 0.2s;
}

#searchInput:focus {
outline: none;
border-color: #667eea;
}

.search-box button {
padding: 15px 40px;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
border-radius: 12px;
font-weight: 700;
cursor: pointer;
font-size: 1rem;
transition: transform 0.2s;
}

.search-box button:hover {
transform: scale(1.05);
}

/* Autocomplete Dropdown */
.autocomplete-suggestions {
position: absolute;
top: 100%;
left: 0;
```

```
right: 0;
background: white;
border: 1px solid #ddd;
border-radius: 8px;
max-height: 300px;
overflow-y: auto;
z-index: 1000;
display: none;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
margin-top: 5px;
}

.autocomplete-suggestions.active {
display: block;
}

.suggestion-item {
padding: 12px 20px;
cursor: pointer;
border-bottom: 1px solid #f0f0f0;
transition: background-color 0.2s;
}

.suggestion-item:last-child {
border-bottom: none;
}

.suggestion-item:hover {
background-color: #f9f9f9;
}

.suggestion-item.highlighted {
background-color: #f0f0f0;
}
```

```
.suggestion-text {
font-size: 0.95rem;
color: #333;
}

.suggestion-label {
font-size: 0.75rem;
color: #999;
margin-left: 8px;
font-weight: 500;
}

@media (max-width: 768px) {
.products-grid {
grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
gap: 15px;
padding: 15px;
}

.filter-section {
margin: 15px;
padding: 15px;
}

.filters-container {
grid-template-columns: 1fr;
}
}

/* Updated Product Card Styles */

.best-price-banner {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
```

```
padding: 16px 15px;
border-radius: 12px;
margin-bottom: 12px;
text-align: center;
}

.best-price-banner .label {
font-size: 0.75rem;
opacity: 0.9;
font-weight: 500;
}

.best-price-banner .price {
font-size: 1.5rem;
font-weight: 700;
margin: 4px 0;
}

/* Store Prices - Clean Minimal Style */
.store-prices {
display: flex;
flex-direction: column;
gap: 10px;
margin-bottom: 15px;
}

.store-price-card {
background: white;
border: 2px solid #e0e0e0;
border-radius: 10px;
padding: 12px 15px;
display: flex;
justify-content: space-between;
align-items: center;
transition: all 0.2s;
```

```
}

.store-price-card:hover {
border-color: #667eea;
box-shadow: 0 2px 8px rgba(102, 126, 234, 0.1);
}

.store-price-card.best-deal {
background: #d4f4dd;
border-color: #7ed957;
}

.store-price-card.unavailable {
opacity: 0.5;
background: #f9f9f9;
}

.store-price-card .store-name {
font-weight: 600;
color: #333;
font-size: 0.95rem;
display: flex;
align-items: center;
}

.store-price-card .price {
font-weight: 700;
color: #111;
font-size: 1rem;
}

/* Add to Cart Button */
.add-to-cart-btn {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
```

```
color: white;
border: none;
padding: 12px 15px;
border-radius: 10px;
font-size: 0.95rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
margin-top: auto;
display: flex;
align-items: center;
justify-content: center;
gap: 6px;
}

.add-to-cart-btn:hover {
transform: translateY(-2px);
}

.quantity-input {
width: 100%;
padding: 10px;
border: 2px solid #e0e0e0;
border-radius: 8px;
text-align: center;
font-size: 0.95rem;
font-weight: 600;
margin-bottom: 10px;
transition: border-color 0.2s;
}

.quantity-input:focus {
outline: none;
border-color: #667eea;
}
```

```
</style>
</head>
<body>
<header class="nav">
<!-- Brand -->
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<!-- Hamburger -->
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<!-- Navigation Links -->
<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<!-- Right Icons -->
<div class="nav-right">
<!-- Notifications -->
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-
6.32V4c0-.83-.67-1.5-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2
2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
```

```
</a>

<!-- Cart -->
<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6
7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-
.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48
0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-
.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>

<!-- Hero Section -->
<section class="hero">
<h1>Search. Compare. Save.</h1>
<p class="tagline">Find the cheapest price across top Kenyan stores instantly.</p>

<form class="search-container" id="searchForm" method="GET" action="{{
url_for('home') }}">
<div class="search-box">
<div class="search-input-wrapper">
<input
type="text"
id="searchInput"
name="search"
placeholder="Search product…"
value="{{ query or '' }}"
autocomplete="off"
>
```

```
<div class="autocomplete-suggestions" id="suggestions"></div>
</div>
<button type="submit">Search</button>
</div>
</form>
</section>

<!-- Filter Section (Only show if searching) -->
{% if query %}
<section class="filter-section">
<h3 class="filter-title">🔍 Filter Results</h3>

<form method="GET" action="{{ url_for('home') }}" id="filterForm">
<input type="hidden" name="search" value="{{ query or '' }}">

<div class="filters-container">
<div class="filter-group">
<label for="min_price">Min Price (KSh)</label>
<input
type="number"
id="min_price"
name="min_price"
placeholder="e.g., 100"
value="{{ min_price or '' }}"
min="0"
step="50"
>
</div>

<div class="filter-group">
<label for="max_price">Max Price (KSh)</label>
<input
type="number"
id="max_price"
name="max_price"
```

```
placeholder="e.g., 5000"
value="{{ max_price or '' }}"
min="0"
step="50"
>
</div>

<div class="filter-group">
<label for="min_discount">Min Discount (%)</label>
<input
type="number"
id="min_discount"
name="min_discount"
placeholder="e.g., 10"
value="{{ min_discount or '' }}"
min="0"
max="100"
step="5"
>
</div>

<div class="filter-actions">
<button type="submit" class="filter-btn filter-btn-apply">
Apply Filters
</button>
<button
type="reset"
class="filter-btn filter-btn-reset"
onclick="clearFilters()"
>
Reset
</button>
</div>
</div>
```

```
<!-- Display Active Filters -->
{% if min_price or max_price or min_discount %}
<div class="active-filters">
{% if min_price %}
<div class="filter-chip">
Min: KSh {{ min_price }}
<button                    onclick="document.getElementById('min_price').value='';
filterForm.submit();">×</button>
</div>
{% endif %}

{% if max_price %}
<div class="filter-chip">
Max: KSh {{ max_price }}
<button                    onclick="document.getElementById('max_price').value='';
filterForm.submit();">×</button>
</div>
{% endif %}

{% if min_discount %}
<div class="filter-chip">
Discount: {{ min_discount }}%+
<button                 onclick="document.getElementById('min_discount').value='';
filterForm.submit();">×</button>
</div>
{% endif %}
</div>
{% endif %}
</form>
</section>
{% endif %}

<!-- Product Results -->
<section class="result">
{% if query %}
```

```
{% if products and products|length > 0 %}
<h2 class="sec-title">Search Results ({{ products|length }})</h2>

<div class="products-grid">
{% for product in products %}
<div class="product-card">
<!-- Product Image -->
<div class="product-image-container">
{% if product['image_url'] %}
<img src="{{ get_image_url(product['image_url']) }}" alt="{{ product['name'] }}"
class="product-image">
{% else %}
<div style="color: #ccc; font-size: 3rem;">🛍️</div>
{% endif %}
</div>

<!-- Product Title -->
<h3>{{ product['name'] }}</h3>
<p class="product-category">{{ product['category'] }}</p>


<!-- Best Price Banner -->
{% if product['best_price'] %}
<div class="best-price-banner">
<div class="label">Best Price:</div>
<div class="price">KSh {{ "{:,.0f}".format(product['best_price']) }}</div>
</div>
{% endif %}

<!-- Store Prices Section -->
<div class="store-prices">
{% for store in ['Carrefour', 'Naivas', 'Quickmart'] %}
{% set store_data = product['stores'].get(store) %}
```

```
<div class="store-price-card {% if store == product['best_store'] %}best-deal{% endif %} {% if not store_data or not store_data.get('current') %}unavailable{% endif %}">
<div class="store-name">
{{ store }}
{% if store == product['best_store'] %}
<span style="background: #4caf50; color: white; padding: 3px 10px; border-radius: 6px; font-size: 0.7rem; margin-left: 6px; font-weight: 700; letter-spacing: 0.5px;">BEST</span>
{% endif %}
</div>
<div class="price">
{% if store_data and store_data.get('current') %}
KSh {{ "{:,.0f}".format(store_data['current']) }}
{% else %}
<span style="color: #999; font-weight: 400;">N/A</span>
{% endif %}
</div>
</div>
{% endfor %}
</div>

<!-- Add to Cart Form -->
<form method="POST" action="{{ url_for('cart_add', product_name=product['name']) }}">
<input type="number" name="quantity" value="1" min="1" class="quantity-input">
<button type="submit" class="add-to-cart-btn">🛒 Add to Cart</button>
</form>
</div>
{% endfor %}
</div>

{% else %}
<div class="no-results">
<p>❌ No products found for "{{ query }}".</p>
```

```
<p>Try searching for a different product or browse by category.</p>
</div>
{% endif %}
{% else %}
<div class="welcome-message">
<h2>👋 Welcome to BeiRadar!</h2>
<p>Search for any product to compare prices across Carrefour, Naivas, and Quickmart.</p>
</div>
{% endif %}
</section>

<footer>
<p>© 2025 BeiRadar</p>
</footer>

</body>


<!-- Update the autocomplete JavaScript in your index.html -->

<script>

const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');

// Animate hamburger into X
hamburger.classList.toggle('open');
});
const searchInput = document.getElementById('searchInput');
const suggestionsBox = document.getElementById('suggestions');
const searchForm = document.getElementById('searchForm');
```

```
let highlightedIndex = -1;

searchInput.addEventListener('input', async (e) => {
const query = e.target.value.trim();

if (query.length < 2) {
suggestionsBox.classList.remove('active');
suggestionsBox.innerHTML = '';
highlightedIndex = -1;
return;
}

try {
const response = await fetch(`/api/search-
suggestions?q=${encodeURIComponent(query)}`);
const data = await response.json();

if (data.suggestions && data.suggestions.length > 0) {
suggestionsBox.innerHTML = data.suggestions.map((suggestion, index) => {
const icon = suggestion.type === 'category' ? '📁' : '🔍';
const label = suggestion.type === 'category' ? 'Category' : 'Product';
return `
<div class="suggestion-item"
onclick="selectSuggestion('${suggestion.text.replace(/'/g, "\\'")}')">
<span class="suggestion-text">${icon} ${suggestion.text}</span>
<span class="suggestion-label">${label}</span>
</div>
`;
}).join('');
suggestionsBox.classList.add('active');
highlightedIndex = -1;
} else {
suggestionsBox.innerHTML = `
<div class="suggestion-item" style="color: #999; cursor: default;">
No products or categories found
```

```
</div>
`;
suggestionsBox.classList.add('active');
}
} catch (error) {
console.error('Error fetching suggestions:', error);
}
});

searchInput.addEventListener('keydown', (e) => {
const items = suggestionsBox.querySelectorAll('.suggestion-item');

if (e.key === 'ArrowDown') {
e.preventDefault();
highlightedIndex = Math.min(highlightedIndex + 1, items.length - 1);
updateHighlight(items);
} else if (e.key === 'ArrowUp') {
e.preventDefault();
highlightedIndex = Math.max(highlightedIndex - 1, -1);
updateHighlight(items);
} else if (e.key === 'Enter') {
e.preventDefault();
if (highlightedIndex >= 0 && items[highlightedIndex]) {
const text = items[highlightedIndex].querySelector('.suggestion-
text').textContent
.replace('🔍 ', '')
.replace('💼 ', '');
selectSuggestion(text);
} else {
searchForm.submit();
}
} else if (e.key === 'Escape') {
suggestionsBox.classList.remove('active');
suggestionsBox.innerHTML = '';
highlightedIndex = -1;
```

```
}
});

function updateHighlight(items) {
items.forEach((item, index) => {
if (index === highlightedIndex) {
item.classList.add('highlighted');
} else {
item.classList.remove('highlighted');
}
});
}

function selectSuggestion(text) {
searchInput.value = text;
suggestionsBox.classList.remove('active');
suggestionsBox.innerHTML = '';
highlightedIndex = -1;
searchForm.submit();
}

document.addEventListener('click', (e) => {
if (e.target !== searchInput) {
suggestionsBox.classList.remove('active');
}
});
</script>

</html>
```

7. categories.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – Categories</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
<!-- Decorative Circles -->
<div class="circle c1"></div>
<div class="circle c2"></div>

<!-- Navigation -->
<header class="nav">
<!-- Brand -->
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<!-- Hamburger -->
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<!-- Navigation Links -->
<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<!-- Right Icons -->
<div class="nav-right">
<!-- Notifications -->
```

```
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2 2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<!-- Cart -->
<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6 7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48 0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>



<!-- Categories Section -->
<section class="result">
<h2 class="sec-title" style="text-align: center;">Categories</h2>
<p class="tagline" style="text-align: center; margin-bottom: 30px;">
Click a category to explore products and compare prices.
</p>

<div class="compare" style="flex-wrap: wrap; justify-content: center; gap: 20px;">
```

```
{% for category in categories %}
<div class="store"
style="width: 200px; cursor: pointer; text-align: center; border-radius: 10px;
padding: 20px; box-shadow: 0 2px 6px rgba(0,0,0,0.2);"
onclick="window.location.href='/categories/{{ category.slug }}'">
<h3>{{ category.name }}</h3>
{% if category.name in CATEGORIES %}
<ul style="list-style: none; padding: 0; margin: 10px 0 0 0; font-size: 0.9em;
color: #555;">
{% for sub in CATEGORIES[category.name][:2] %}
<li>{{ sub }}</li>
{% endfor %}
{% if CATEGORIES[category.name]|length > 2 %}
<li>…</li>
{% endif %}
</ul>
{% endif %}
</div>
{% endfor %}
</div>
</section>

</body>
<script>
const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});
</script>

</html>
```

8. category_detail.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – {{ category_name }}</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Card container */
.subcategory-container {
display: flex;
flex-wrap: wrap;
justify-content: center;
gap: 25px;
margin-top: 30px;
}

/* Individual cards */
.subcategory-card {
width: 160px;
height: 160px;
background: linear-gradient(145deg, #fef6e4, #fff9f0);
border-radius: 16px;
box-shadow: 0 6px 15px rgba(0,0,0,0.08);
display: flex;
flex-direction: column;
justify-content: center;
align-items: center;
text-align: center;
font-size: 1.1rem;
cursor: pointer;
```

```
transition: all 0.3s ease;
position: relative;
}

.subcategory-card::before {
content: '';
position: absolute;
top: -10px;
left: -10px;
width: calc(100% + 20px);
height: calc(100% + 20px);
border-radius: 16px;
background: linear-gradient(45deg, #ffd5d5, #ffe5a8, #c4f0ff);
z-index: -1;
opacity: 0;
transition: opacity 0.3s ease;
}

.subcategory-card:hover::before {
opacity: 1;
}

.subcategory-card:hover {
transform: translateY(-8px) scale(1.05);
box-shadow: 0 12px 20px rgba(0,0,0,0.15);
}

/* Icon inside card */
.subcategory-card span.icon {
font-size: 3rem;
margin-bottom: 10px;
}

.subcategory-card h4 {
margin: 0;
```

```
font-weight: 600;
color: #333;
}

h2.sec-title {
margin-top: 30px;
font-size: 2rem;
text-align: center;
}

.tagline {
text-align: center;
margin-bottom: 40px;
font-size: 1rem;
color: #555;
}
</style>
</head>

<body>

<body>
<!-- Decorative Circles -->
<div class="circle c1"></div>
<div class="circle c2"></div>

<!-- Navigation -->
<header class="nav">
<!-- Brand -->
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<!-- Hamburger -->
```

```
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<!-- Navigation Links -->
<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<!-- Right Icons -->
<div class="nav-right">
<!-- Notifications -->
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-
6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2
2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<!-- Cart -->
<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6
7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-
.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48
0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-
.9-2-2-2z"/>
</svg>
```

```
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>
</body>

<!-- Subcategories Section -->
<section class="result">
<h2 class="sec-title">{{ category_name }}</h2>
<p class="tagline">Click a subcategory to explore products and compare prices.</p>

<div class="subcategory-container">
{% for sub in subcategories %}
<div class="subcategory-card"
onclick="window.location.href='{{              url_for('products_by_subcategory',
subcategory_slug=sub|lower|replace(' ', '-')) }}'">
<span class="icon">
{% if sub == 'Rice' %}&#127834;
{% elif sub == 'Cooking Oil' %}&#x1F373;
{% elif sub == 'Sugar' %}&#127852;
{% elif sub == 'Milk' %}&#129371;
{% elif sub == 'Yoghurt' %}&#129371;
{% elif sub == 'Cheese' %}&#129472;
{% elif sub == 'Cleaning Products' %}&#129508;
{% elif sub == 'Soap' %}&#129484;
{% elif sub == 'Toilet Paper' %}&#129531;
{% elif sub == 'Toothpaste' %}&#128290;
{% elif sub == 'Lotion' %}&#128717;
{% elif sub == 'Sanitary Pads' %}&#10084;
{% else %}&#128717;{% endif %}
</span>
<h4>{{ sub }}</h4>
</div>
```

```
{% endfor %}
</div>

</section>

</body>
<script>
const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});
</script>

</html>
```

9. categories_list.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – Categories</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Main category cards */
.category-container {
display: flex;
flex-wrap: wrap;
```

```
justify-content: center;
gap: 25px;
margin-top: 30px;
}

.category-card {
width: 200px;
height: 200px;
background-color: #ffffff;
border-radius: 12px;
box-shadow: 0 3px 10px rgba(0,0,0,0.2);
display: flex;
flex-direction: column;
justify-content: center;
align-items: center;
cursor: pointer;
font-size: 1.2em;
transition: transform 0.2s, box-shadow 0.2s;
}

.category-card:hover {
transform: translateY(-5px);
box-shadow: 0 5px 15px rgba(0,0,0,0.3);
}

.category-emoji {
font-size: 3em;
margin-bottom: 10px;
}

</style>
</head>
<body>
<header class="nav">
<!-- Brand Logo -->
```

```
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<!-- Hamburger Menu -->
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<!-- Navigation Links -->
<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<!-- Right Side Icons -->
<div class="nav-right">
<!-- Notifications Icon -->
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-
6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2
2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<!-- Shopping Cart Icon -->
<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
```

```
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6
7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-
.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48
0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-
.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>


<!-- Categories Section -->
<section class="result">
<h2 class="sec-title" style="text-align: center;">Categories</h2>
<p class="tagline" style="text-align: center; margin-bottom: 30px;">
Click a category to explore subcategories!
</p>

<div class="category-container">
{% for category in categories %}
<div      class="category-card"      onclick="window.location.href='/categories/{{
category.slug }}'">
<div class="category-emoji">
{% if category.name == 'Foodstuff' %}🍚{% elif category.name == 'Dairy' %}🥛{% elif
category.name == 'Household' %}🧹{% elif category.name == 'Personal Care' %}🧴{%
else %}🛍️{% endif %}
</div>
<div>{{ category.name }}</div>
</div>
{% endfor %}
</div>
</section>
```

```
</body>
</html>
```

10. category_products.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – {{ category_name }}</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Product Grid Layout */
.products-grid {
display: grid;
grid-template-columns: repeat(auto-fill, minmax(350px, 1fr));
gap: 30px;
padding: 30px;
max-width: 1400px;
margin: 0 auto;
}

/* Product Card - Similar to first image */
.product-card {
background: white;
border-radius: 16px;
padding: 25px;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
transition: transform 0.2s, box-shadow 0.2s;
display: flex;
flex-direction: column;
```

```
align-items: center;
}

.product-card:hover {
transform: translateY(-5px);
box-shadow: 0 8px 20px rgba(0, 0, 0, 0.12);
}

/* Product Image - Centered */
.product-image-container {
width: 100%;
height: 200px;
display: flex;
align-items: center;
justify-content: center;
margin-bottom: 20px;
background: #f9f9f9;
border-radius: 12px;
padding: 15px;
}

.product-image {
max-width: 100%;
max-height: 100%;
object-fit: contain;
}

/* Product Title */
.product-card h3 {
font-size: 1.1rem;
font-weight: 600;
color: #111;
margin: 0 0 8px 0;
text-align: center;
line-height: 1.4;
```

```
}

/* Category Label */
.product-category {
color: #666;
font-size: 0.9rem;
margin-bottom: 15px;
text-align: center;
}

/* Best Price Banner - Like first image */
.best-price-banner {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 12px 15px;
border-radius: 12px;
margin-bottom: 15px;
display: flex;
justify-content: space-between;
align-items: center;
}

.best-price-banner .price-info {
display: flex;
flex-direction: column;
gap: 4px;
}

.best-price-banner .label {
font-size: 0.75rem;
opacity: 0.9;
}

.best-price-banner .price {
```

```
font-size: 1.3rem;
font-weight: 700;
}

.best-price-banner .on-sale-badge {
background: #ff6b6b;
padding: 6px 12px;
border-radius: 20px;
font-size: 0.75rem;
font-weight: 600;
}

/* Store Comparison Section */
.store-comparison {
width: 100%;
margin-bottom: 15px;
}

.best-deal-tag {
text-align: center;
background: linear-gradient(135deg, #4facfe 0%, #00f2fe 100%);
color: white;
padding: 8px 15px;
border-radius: 20px;
font-size: 0.85rem;
font-weight: 600;
margin-bottom: 10px;
}

.store-prices {
display: flex;
flex-direction: column;
gap: 10px;
}
```

```css
.store-price-card {
background: #f9f9f9;
border: 2px solid transparent;
border-radius: 10px;
padding: 12px 15px;
display: flex;
justify-content: space-between;
align-items: center;
transition: all 0.2s;
}

.store-price-card:hover {
background: #fff;
border-color: #667eea;
}

.store-price-card.best-deal {
background: linear-gradient(135deg, #d4fc79 0%, #96e6a1 100%);
border-color: #4caf50;
}

.store-price-card .store-name {
font-weight: 600;
color: #333;
font-size: 0.95rem;
}

.store-price-card .price-section {
text-align: right;
}

.store-price-card .current-price {
font-size: 1.1rem;
font-weight: 700;
color: #111;
```

```
margin: 0;
}

.store-price-card .original-price {
font-size: 0.8rem;
color: #999;
text-decoration: line-through;
margin: 0;
}

.store-price-card .discount-label {
font-size: 0.75rem;
color: #ff6b6b;
font-weight: 600;
}

/* Add to Cart Section */
.add-to-cart-section {
width: 100%;
display: flex;
gap: 10px;
align-items: center;
justify-content: center;
margin-top: 15px;
}

.quantity-input {
width: 60px;
padding: 8px;
border: 2px solid #ddd;
border-radius: 8px;
text-align: center;
font-size: 1rem;
font-weight: 600;
}
```

```css
.add-to-cart-btn {
flex: 1;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
padding: 10px 20px;
border-radius: 8px;
font-size: 1rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
}

.add-to-cart-btn:hover {
transform: scale(1.05);
}

/* Section Title */
.sec-title {
text-align: center;
font-size: 2rem;
font-weight: 700;
color: #111;
margin: 30px 0;
}

/* No Products Message */
.no-products {
text-align: center;
padding: 50px;
color: #666;
font-size: 1.1rem;
}
```

```
/* Responsive */
@media (max-width: 768px) {
.products-grid {
grid-template-columns: 1fr;
padding: 15px;
}

.best-price-banner {
flex-direction: column;
gap: 10px;
text-align: center;
}
}
</style>
</head>
<body>
<header class="nav">
<!-- Brand -->
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<!-- Hamburger -->
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<!-- Navigation Links -->
<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
```

```
<a href="{{ url_for('about') }}">About</a>
</nav>

<!-- Right Icons -->
<div class="nav-right">
<!-- Notifications -->
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2 2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<!-- Cart -->
<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6 7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48 0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>

<!-- Products Section -->
<section class="result">
<h2 class="sec-title">{{ category_name }}</h2>
```

```
{% if products %}
<div class="products-grid">
{% for product in products %}
<div class="product-card">
<!-- Product Image - Centered -->
<div class="product-image-container">
{% if product['image_url'] %}
<img src="{{ get_image_url(product['image_url']) }}" alt="{{ product['product'] }}" width="200">


{% else %}
<div style="color: #ccc; font-size: 3rem;">🛍️</div>
{% endif %}

</div>

<!-- Product Title -->
<h3>{{ product['name'] }}</h3>

<!-- Category Label -->
{% if product.get('category') %}
<p class="product-category">{{ product['category'] }}</p>
{% endif %}

<!-- Best Price Banner -->
{% if product['best_price'] %}
<div class="best-price-banner">
<div class="price-info">
<span class="label">Best Price:</span>
<span class="price">KSh {{ "{:,.0f}".format(product['best_price']) }}</span>
</div>
{% if product.get('on_sale') %}
<div class="on-sale-badge">On Sale</div>
{% endif %}
```

```
</div>
{% endif %}


<!-- Store Comparison -->
<div class="store-comparison">
{% if product['best_store'] %}
<div class="best-deal-tag">✓ Best Deal</div>
{% endif %}

<div class="store-prices">
{% for store_name in ['Carrefour', 'Naivas', 'Quickmart'] %}
{% set store_data = product['stores'].get(store_name) %}
<div class="store-price-card {% if store_name == product['best_store'] %}best-deal{% endif %}">
<div class="store-name">
{{ store_name }}
{% if store_name == product['best_store'] %}
<span style="background: #4caf50; color: white; padding: 2px 8px; border-radius: 4px; font-size: 0.75rem; margin-left: 8px; font-weight: 600;">✓ BEST</span>
{% endif %}
</div>
<div class="price-section">
{% if store_data and store_data.get('current') %}
<p class="current-price">KSh {{ "{:,.0f}".format(store_data['current']) }}</p>
{% if store_data.get('original') and store_data['original'] != store_data['current'] %}
<p class="original-price">KSh {{ "{:,.0f}".format(store_data['original']) }}</p>
<p class="discount-label">
Save KSh {{ "{:,.0f}".format(store_data['original'] - store_data['current']) }}
</p>
{% endif %}
{% else %}
<p class="current-price" style="color: #999; font-weight: 400;">Not Available</p>
{% endif %}
```

```
</div>
</div>
{% endfor %}
</div>
</div>

<!-- Add to Cart Form -->
<form method="POST" action="{{ url_for('cart_add', product_name=product['name'])
}}">
<input type="number" name="quantity" value="1" min="1" class="quantity-input">
<button type="submit" class="add-to-cart-btn">🛒 Add to Cart</button>
</form>
</div>
{% endfor %}
</div>
{% else %}
<div class="no-products">
<p>Click a subcategory to explore products and compare prices.</p>
</div>
{% endif %}
</section>
</body>
<script>
const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});
```

```
</script>
</html>
```

11. subcategory_products.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – {{ category_name }}</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Product Grid Layout */
.products-grid {
display: grid;
grid-template-columns: repeat(auto-fill, minmax(350px, 1fr));
gap: 30px;
padding: 30px;
max-width: 1400px;
margin: 0 auto;
}

/* Product Card - Similar to first image */
.product-card {
background: white;
border-radius: 16px;
padding: 25px;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
transition: transform 0.2s, box-shadow 0.2s;
display: flex;
flex-direction: column;
align-items: center;
}
```

```css
.product-card:hover {
transform: translateY(-5px);
box-shadow: 0 8px 20px rgba(0, 0, 0, 0.12);
}

/* Product Image - Centered */
.product-image-container {
width: 100%;
height: 200px;
display: flex;
align-items: center;
justify-content: center;
margin-bottom: 20px;
background: #f9f9f9;
border-radius: 12px;
padding: 15px;
}

.product-image {
max-width: 100%;
max-height: 100%;
object-fit: contain;
}

/* Product Title */
.product-card h3 {
font-size: 1.1rem;
font-weight: 600;
color: #111;
margin: 0 0 8px 0;
text-align: center;
line-height: 1.4;
}
```

```css
/* Category Label */
.product-category {
color: #666;
font-size: 0.9rem;
margin-bottom: 15px;
text-align: center;
}

/* Best Price Banner - Like first image */
.best-price-banner {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 12px 15px;
border-radius: 12px;
margin-bottom: 15px;
display: flex;
justify-content: space-between;
align-items: center;
}

.best-price-banner .price-info {
display: flex;
flex-direction: column;
gap: 4px;
}

.best-price-banner .label {
font-size: 0.75rem;
opacity: 0.9;
}

.best-price-banner .price {
font-size: 1.3rem;
font-weight: 700;
```

```
}

.best-price-banner .on-sale-badge {
background: #ff6b6b;
padding: 6px 12px;
border-radius: 20px;
font-size: 0.75rem;
font-weight: 600;
}

/* Store Comparison Section */
.store-comparison {
width: 100%;
margin-bottom: 15px;
}

.best-deal-tag {
text-align: center;
background: linear-gradient(135deg, #4facfe 0%, #00f2fe 100%);
color: white;
padding: 8px 15px;
border-radius: 20px;
font-size: 0.85rem;
font-weight: 600;
margin-bottom: 10px;
}

.store-prices {
display: flex;
flex-direction: column;
gap: 10px;
}

.store-price-card {
background: #f9f9f9;
```

```
border: 2px solid transparent;
border-radius: 10px;
padding: 12px 15px;
display: flex;
justify-content: space-between;
align-items: center;
transition: all 0.2s;
}

.store-price-card:hover {
background: #fff;
border-color: #667eea;
}

.store-price-card.best-deal {
background: linear-gradient(135deg, #d4fc79 0%, #96e6a1 100%);
border-color: #4caf50;
}

.store-price-card .store-name {
font-weight: 600;
color: #333;
font-size: 0.95rem;
}

.store-price-card .price-section {
text-align: right;
}

.store-price-card .current-price {
font-size: 1.1rem;
font-weight: 700;
color: #111;
margin: 0;
}
```

```
.store-price-card .original-price {
font-size: 0.8rem;
color: #999;
text-decoration: line-through;
margin: 0;
}

.store-price-card .discount-label {
font-size: 0.75rem;
color: #ff6b6b;
font-weight: 600;
}

/* Add to Cart Section */
.add-to-cart-section {
width: 100%;
display: flex;
gap: 10px;
align-items: center;
justify-content: center;
margin-top: 15px;
}

.quantity-input {
width: 60px;
padding: 8px;
border: 2px solid #ddd;
border-radius: 8px;
text-align: center;
font-size: 1rem;
font-weight: 600;
}

.add-to-cart-btn {
```

```
flex: 1;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
padding: 10px 20px;
border-radius: 8px;
font-size: 1rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
}

.add-to-cart-btn:hover {
transform: scale(1.05);
}

/* Section Title */
.sec-title {
text-align: center;
font-size: 2rem;
font-weight: 700;
color: #111;
margin: 30px 0;
}

/* No Products Message */
.no-products {
text-align: center;
padding: 50px;
color: #666;
font-size: 1.1rem;
}

/* Responsive */
@media (max-width: 768px) {
```

```
.products-grid {
grid-template-columns: 1fr;
padding: 15px;
}

.best-price-banner {
flex-direction: column;
gap: 10px;
text-align: center;
}
}
</style>
</head>
<body>
<header class="nav">
<!-- Brand -->
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<!-- Hamburger -->
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<!-- Navigation Links -->
<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>
```

```
<!-- Right Icons -->
<div class="nav-right">
<!-- Notifications -->
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2 2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<!-- Cart -->
<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6 7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48 0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>

<!-- Products Section -->
<section class="result">
<h2 class="sec-title">{{ category_name }}</h2>

{% if products %}
<div class="products-grid">
```

```
{% for product in products %}
<div class="product-card">
<!-- Product Image - Centered -->
<div class="product-image-container">
{% if product['image_url'] %}
<img src="{{ get_image_url(product['image_url']) }}" alt="{{ product['product'] }}" width="200">


{% else %}
<div style="color: #ccc; font-size: 3rem;">🛍️</div>
{% endif %}

</div>

<!-- Product Title -->
<h3>{{ product['name'] }}</h3>

<!-- Category Label -->
{% if product.get('category') %}
<p class="product-category">{{ product['category'] }}</p>
{% endif %}

<!-- Best Price Banner -->
{% if product['best_price'] %}
<div class="best-price-banner">
<div class="price-info">
<span class="label">Best Price:</span>
<span class="price">KSh {{ "{:,.0f}".format(product['best_price']) }}</span>
</div>
{% if product.get('on_sale') %}
<div class="on-sale-badge">On Sale</div>
{% endif %}
</div>
{% endif %}
```

```
<!-- Store Comparison -->
<div class="store-comparison">
{% if product['best_store'] %}
<div class="best-deal-tag">✓ Best Deal</div>
{% endif %}

<div class="store-prices">
{% for store_name in ['Carrefour', 'Naivas', 'Quickmart'] %}
{% set store_data = product['stores'].get(store_name) %}
<div class="store-price-card {% if store_name == product['best_store'] %}best-deal{% endif %}">
<div class="store-name">
{{ store_name }}
{% if store_name == product['best_store'] %}
<span style="background: #4caf50; color: white; padding: 2px 8px; border-radius: 4px; font-size: 0.75rem; margin-left: 8px; font-weight: 600;">✓ BEST</span>
{% endif %}
</div>
<div class="price-section">
{% if store_data and store_data.get('current') %}
<p class="current-price">KSh {{ "{:,.0f}".format(store_data['current']) }}</p>
{% if store_data.get('original') and store_data['original'] != store_data['current'] %}
<p class="original-price">KSh {{ "{:,.0f}".format(store_data['original']) }}</p>
<p class="discount-label">
Save KSh {{ "{:,.0f}".format(store_data['original'] - store_data['current']) }}
</p>
{% endif %}
{% else %}
<p class="current-price" style="color: #999; font-weight: 400;">Not Available</p>
{% endif %}
</div>
</div>
```

```
{% endfor %}
</div>
</div>

<!-- Add to Cart Form -->
<form method="POST" action="{{ url_for('cart_add', product_name=product['name'])
}}">
<input type="number" name="quantity" value="1" min="1" class="quantity-input">
<button type="submit" class="add-to-cart-btn">🛒 Add to Cart</button>
</form>
</div>
{% endfor %}
</div>
{% else %}
<div class="no-products">
<p>Click a subcategory to explore products and compare prices.</p>
</div>
{% endif %}
</section>
</body>
<script>
const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});
</script>
</html>
```

12. results.html

```
<div class="results-container">
{% for product in products %}
<div class="result-card">
<img src="{{ product.image }}" class="product-image">

<div class="result-details">
<h3 class="product-name">{{ product.name }}</h3>

<div class="price-row">
<span class="price">KES {{ product.price }}</span>

{% if product.is_cheapest %}
<span class="badge-cheapest">Cheapest</span>
{% endif %}
</div>

<div class="shop-row">
<img src="/static/icons/shop.svg" class="shop-icon">
<span class="shop-name">{{ product.store }}</span>
</div>
</div>
</div>
{% endfor %}
</div>
```

12. deals.html

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – Hot Deals</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Deals Hero Section */
.deals-hero {
background: white;
padding: 60px 20px;
text-align: center;
margin-bottom: 30px;
margin-top: 30px;
border-bottom: 1px solid #eee;
}

.deals-hero h1 {
font-size: 2.5rem;
font-weight: 700;
color: #111;
margin-bottom: 15px;
}

.deals-hero p {
font-size: 1rem;
color: #666;
margin-bottom: 20px;
}

.deals-count {
display: inline-block;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 10px 20px;
```

```
border-radius: 20px;
font-size: 0.9rem;
font-weight: 600;
}

/* Store Filter Section */
.store-filter {
display: flex;
justify-content: center;
gap: 12px;
flex-wrap: wrap;
margin-top: 20px;
}

.store-filter-btn {
background: #f5f5f5;
color: #333;
border: 2px solid #ddd;
padding: 10px 18px;
border-radius: 20px;
cursor: pointer;
font-weight: 600;
font-size: 0.9rem;
transition: all 0.2s;
}

.store-filter-btn:hover {
background: #efefef;
border-color: #667eea;
}

.store-filter-btn.active {
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border-color: #667eea;
```

```
}

/* Deals Grid */
.deals-grid {
display: grid;
grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
gap: 25px;
padding: 30px;
max-width: 1400px;
margin: 0 auto;
}

.deal-card {
background: white;
border-radius: 16px;
overflow: hidden;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
transition: transform 0.2s, box-shadow 0.2s;
display: flex;
flex-direction: column;
}

.deal-card:hover {
transform: translateY(-5px);
box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
}

/* Discount Badge */
.discount-badge {
position: absolute;
top: 15px;
right: 15px;
background: #ff6b6b;
color: white;
padding: 8px 12px;
```

```
border-radius: 8px;
font-weight: 700;
font-size: 0.85rem;
z-index: 10;
}

/* Product Image */
.deal-image-container {
width: 100%;
height: 200px;
display: flex;
align-items: center;
justify-content: center;
background: #f9f9f9;
padding: 15px;
position: relative;
}

.deal-image {
max-width: 100%;
max-height: 100%;
object-fit: contain;
}

/* Deal Content */
.deal-content {
padding: 20px;
flex: 1;
display: flex;
flex-direction: column;
}

.deal-product-name {
font-size: 1rem;
font-weight: 700;
```

```
color: #111;
margin: 0 0 8px 0;
line-height: 1.3;
}

.deal-category {
color: #666;
font-size: 0.85rem;
margin-bottom: 10px;
}

.deal-store-badge {
display: inline-block;
background: #667eea;
color: white;
padding: 6px 12px;
border-radius: 6px;
font-size: 0.8rem;
font-weight: 600;
margin-bottom: 12px;
width: fit-content;
}

/* Pricing Section */
.deal-pricing {
background: #f9f9f9;
padding: 15px;
border-radius: 12px;
margin-top: auto;
margin-bottom: 15px;
}

.price-row {
display: flex;
justify-content: space-between;
```

```
align-items: center;
margin-bottom: 10px;
}

.price-row:last-child {
margin-bottom: 0;
}

.price-label {
color: #666;
font-size: 0.9rem;
font-weight: 600;
}

.original-price {
font-size: 1rem;
color: #999;
text-decoration: line-through;
}

.deal-price {
font-size: 1.4rem;
font-weight: 700;
color: #4caf50;
}

.savings {
font-size: 0.95rem;
color: #ff6b6b;
font-weight: 600;
padding-top: 10px;
border-top: 1px solid #eee;
margin-top: 10px;
}
```

```css
/* Add to Cart Button */
.deal-add-btn {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
padding: 12px 15px;
border-radius: 10px;
font-size: 0.95rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
}

.deal-add-btn:hover {
transform: scale(1.05);
}

/* No Deals */
.no-deals {
text-align: center;
padding: 80px 20px;
color: #666;
}

.no-deals h2 {
font-size: 1.5rem;
color: #111;
margin-bottom: 10px;
}

.no-deals p {
font-size: 1.05rem;
margin-bottom: 20px;
}
```

```
.no-deals a {
display: inline-block;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 12px 30px;
border-radius: 8px;
text-decoration: none;
font-weight: 600;
}

/* Responsive */
@media (max-width: 768px) {
.deals-hero {
padding: 40px 20px;
}

.deals-hero h1 {
font-size: 1.8rem;
}

.deals-grid {
grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
gap: 15px;
padding: 15px;
}

.discount-badge {
padding: 8px 12px;
font-size: 0.85rem;
}
}
</style>
</head>
```

```
<body>
<header class="nav">
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>

<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<div class="nav-right">
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-
6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2
2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6
7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-
.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48
```

```
0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-
.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>

<!-- Deals Hero Section -->
<section class="deals-hero">
<h1>🔥 Hot Deals</h1>
<p>Get the biggest discounts across Nairobi's top supermarkets</p>
{% if deals %}
<div class="deals-count">{{ deals|length }} Amazing Deals Available</div>
{% endif %}

<!-- Store Filter -->
<div class="store-filter">
<button    class="store-filter-btn    active"    onclick="filterDeals('all')">All
Stores</button>
<button       class="store-filter-btn"       onclick="filterDeals('Carrefour')">🏪
Carrefour</button>
<button        class="store-filter-btn"        onclick="filterDeals('Naivas')">🏪
Naivas</button>
<button       class="store-filter-btn"       onclick="filterDeals('Quickmart')">🏪
Quickmart</button>
</div>
</section>

<!-- Deals Grid Section -->
<section>
{% if deals %}
```

```
<div class="deals-grid">
{% for deal in deals %}
<div class="deal-card" data-store="{{ deal.store }}">
<!-- Discount Badge -->
<div class="discount-badge">
🎉 {{ "{:.0f}".format(deal.deal_percentage) }}% OFF
</div>

<!-- Product Image -->
<div class="deal-image-container">
{% if deal.image_url %}
<img  src="{{  get_image_url(deal.image_url)  }}"  alt="{{  deal.product_name  }}"
class="deal-image">
{% else %}
<div style="color: #ccc; font-size: 3rem;">🛍️</div>
{% endif %}
</div>

<!-- Deal Content -->
<div class="deal-content">
<h3 class="deal-product-name">{{ deal.product_name }}</h3>
<p class="deal-category">{{ deal.category }}</p>

<!-- Store Badge -->
<div class="deal-store-badge">
🏪 {{ deal.store }}
</div>

<!-- Pricing -->
<div class="deal-pricing">
<div class="price-row">
<span class="price-label">Original Price:</span>
<span class="original-price">KSh {{ "{:,.0f}".format(deal.old_price) }}</span>
</div>
<div class="price-row">
```

```
<span class="price-label">Deal Price:</span>
<span class="deal-price">KSh {{ "{:,.0f}".format(deal.new_price) }}</span>
</div>
<div class="savings">
💰 Save KSh {{ "{:,.0f}".format(deal.old_price - deal.new_price) }}
</div>
</div>

<!-- Add to Cart Button -->
<form method="POST" action="{{ url_for('cart_add', product_name=deal.product_name)
}}">
<input type="hidden" name="quantity" value="1">
<button type="submit" class="deal-add-btn">🛒 Add Deal to Cart</button>
</form>
</div>
</div>
{% endfor %}
</div>
{% else %}
<div class="no-deals">
<h2>No Hot Deals Right Now</h2>
<p>Check back later for amazing discounts!</p>
<a href="{{ url_for('home') }}">← Browse All Products</a>
</div>
{% endif %}
</section>

<footer>
<p>© 2025 BeiRadar</p>
</footer>

</body>

<script>
const hamburger = document.getElementById('hamburger');
```

```
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});

function filterDeals(store) {
const dealCards = document.querySelectorAll('.deal-card');
const filterBtns = document.querySelectorAll('.store-filter-btn');

// Update active button
filterBtns.forEach(btn => btn.classList.remove('active'));
event.target.classList.add('active');

// Filter deals
dealCards.forEach(card => {
if (store === 'all' || card.dataset.store === store) {
card.style.display = '';
} else {
card.style.display = 'none';
}
});
}
</script>

</html>
```

13. cart.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – Hot Deals</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
<style>
/* Deals Hero Section */
.deals-hero {
background: white;
padding: 60px 20px;
text-align: center;
margin-bottom: 30px;
margin-top: 30px;
border-bottom: 1px solid #eee;
}

.deals-hero h1 {
font-size: 2.5rem;
font-weight: 700;
color: #111;
margin-bottom: 15px;
}

.deals-hero p {
font-size: 1rem;
color: #666;
margin-bottom: 20px;
}

.deals-count {
display: inline-block;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 10px 20px;
border-radius: 20px;
font-size: 0.9rem;
font-weight: 600;
```

```
}

/* Store Filter Section */
.store-filter {
display: flex;
justify-content: center;
gap: 12px;
flex-wrap: wrap;
margin-top: 20px;
}

.store-filter-btn {
background: #f5f5f5;
color: #333;
border: 2px solid #ddd;
padding: 10px 18px;
border-radius: 20px;
cursor: pointer;
font-weight: 600;
font-size: 0.9rem;
transition: all 0.2s;
}

.store-filter-btn:hover {
background: #efefef;
border-color: #667eea;
}

.store-filter-btn.active {
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border-color: #667eea;
}

/* Deals Grid */
```

```
.deals-grid {
display: grid;
grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
gap: 25px;
padding: 30px;
max-width: 1400px;
margin: 0 auto;
}

.deal-card {
background: white;
border-radius: 16px;
overflow: hidden;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
transition: transform 0.2s, box-shadow 0.2s;
display: flex;
flex-direction: column;
}

.deal-card:hover {
transform: translateY(-5px);
box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
}

/* Discount Badge */
.discount-badge {
position: absolute;
top: 15px;
right: 15px;
background: #ff6b6b;
color: white;
padding: 8px 12px;
border-radius: 8px;
font-weight: 700;
font-size: 0.85rem;
```

```
z-index: 10;
}

/* Product Image */
.deal-image-container {
width: 100%;
height: 200px;
display: flex;
align-items: center;
justify-content: center;
background: #f9f9f9;
padding: 15px;
position: relative;
}

.deal-image {
max-width: 100%;
max-height: 100%;
object-fit: contain;
}

/* Deal Content */
.deal-content {
padding: 20px;
flex: 1;
display: flex;
flex-direction: column;
}

.deal-product-name {
font-size: 1rem;
font-weight: 700;
color: #111;
margin: 0 0 8px 0;
line-height: 1.3;
```

```
}

.deal-category {
color: #666;
font-size: 0.85rem;
margin-bottom: 10px;
}

.deal-store-badge {
display: inline-block;
background: #667eea;
color: white;
padding: 6px 12px;
border-radius: 6px;
font-size: 0.8rem;
font-weight: 600;
margin-bottom: 12px;
width: fit-content;
}

/* Pricing Section */
.deal-pricing {
background: #f9f9f9;
padding: 15px;
border-radius: 12px;
margin-top: auto;
margin-bottom: 15px;
}

.price-row {
display: flex;
justify-content: space-between;
align-items: center;
margin-bottom: 10px;
}
```

```
.price-row:last-child {
margin-bottom: 0;
}

.price-label {
color: #666;
font-size: 0.9rem;
font-weight: 600;
}

.original-price {
font-size: 1rem;
color: #999;
text-decoration: line-through;
}

.deal-price {
font-size: 1.4rem;
font-weight: 700;
color: #4caf50;
}

.savings {
font-size: 0.95rem;
color: #ff6b6b;
font-weight: 600;
padding-top: 10px;
border-top: 1px solid #eee;
margin-top: 10px;
}

/* Add to Cart Button */
.deal-add-btn {
width: 100%;
```

```
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
padding: 12px 15px;
border-radius: 10px;
font-size: 0.95rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
}

.deal-add-btn:hover {
transform: scale(1.05);
}

/* No Deals */
.no-deals {
text-align: center;
padding: 80px 20px;
color: #666;
}

.no-deals h2 {
font-size: 1.5rem;
color: #111;
margin-bottom: 10px;
}

.no-deals p {
font-size: 1.05rem;
margin-bottom: 20px;
}

.no-deals a {
display: inline-block;
```

```
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 12px 30px;
border-radius: 8px;
text-decoration: none;
font-weight: 600;
}

/* Responsive */
@media (max-width: 768px) {
.deals-hero {
padding: 40px 20px;
}

.deals-hero h1 {
font-size: 1.8rem;
}

.deals-grid {
grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
gap: 15px;
padding: 15px;
}

.discount-badge {
padding: 8px 12px;
font-size: 0.85rem;
}
}
</style>
</head>

<body>
<header class="nav">
<div class="brand">
```

```
<span class="brand-icon">📡</span>
BeiRadar
</div>

<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<div class="nav-right">
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2 2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6 7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48 0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-.9-2-2-2z"/>
</svg>
```

```
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
</a>
</div>
</header>

<!-- Deals Hero Section -->
<section class="deals-hero">
<h1>🔥 Hot Deals</h1>
<p>Get the biggest discounts across Nairobi's top supermarkets</p>
{% if deals %}
<div class="deals-count">{{ deals|length }} Amazing Deals Available</div>
{% endif %}

<!-- Store Filter -->
<div class="store-filter">
<button class="store-filter-btn active" onclick="filterDeals('all')">All
Stores</button>
<button class="store-filter-btn" onclick="filterDeals('Carrefour')">🏪
Carrefour</button>
<button class="store-filter-btn" onclick="filterDeals('Naivas')">🏪
Naivas</button>
<button class="store-filter-btn" onclick="filterDeals('Quickmart')">🏪
Quickmart</button>
</div>
</section>

<!-- Deals Grid Section -->
<section>
{% if deals %}
<div class="deals-grid">
{% for deal in deals %}
<div class="deal-card" data-store="{{ deal.store }}">
```

```
<!-- Discount Badge -->
<div class="discount-badge">
🎉 {{ "{:.0f}".format(deal.deal_percentage) }}% OFF
</div>

<!-- Product Image -->
<div class="deal-image-container">
{% if deal.image_url %}
<img  src="{{  get_image_url(deal.image_url)  }}"  alt="{{  deal.product_name  }}"
class="deal-image">
{% else %}
<div style="color: #ccc; font-size: 3rem;">🛍️</div>
{% endif %}
</div>

<!-- Deal Content -->
<div class="deal-content">
<h3 class="deal-product-name">{{ deal.product_name }}</h3>
<p class="deal-category">{{ deal.category }}</p>

<!-- Store Badge -->
<div class="deal-store-badge">
🏪 {{ deal.store }}
</div>

<!-- Pricing -->
<div class="deal-pricing">
<div class="price-row">
<span class="price-label">Original Price:</span>
<span class="original-price">KSh {{ "{:,.0f}".format(deal.old_price) }}</span>
</div>
<div class="price-row">
<span class="price-label">Deal Price:</span>
<span class="deal-price">KSh {{ "{:,.0f}".format(deal.new_price) }}</span>
</div>
```

```
<div class="savings">
💰 Save KSh {{ "{:,.0f}".format(deal.old_price - deal.new_price) }}
</div>
</div>

<!-- Add to Cart Button -->
<form method="POST" action="{{ url_for('cart_add', product_name=deal.product_name)
}}">
<input type="hidden" name="quantity" value="1">
<button type="submit" class="deal-add-btn">🛒 Add Deal to Cart</button>
</form>
</div>
</div>
{% endfor %}
</div>
{% else %}
<div class="no-deals">
<h2>No Hot Deals Right Now</h2>
<p>Check back later for amazing discounts!</p>
<a href="{{ url_for('home') }}">← Browse All Products</a>
</div>
{% endif %}
</section>

<footer>
<p>© 2025 BeiRadar</p>
</footer>

</body>

<script>
const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
```

```
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});

function filterDeals(store) {
const dealCards = document.querySelectorAll('.deal-card');
const filterBtns = document.querySelectorAll('.store-filter-btn');

// Update active button
filterBtns.forEach(btn => btn.classList.remove('active'));
event.target.classList.add('active');

// Filter deals
dealCards.forEach(card => {
if (store === 'all' || card.dataset.store === store) {
card.style.display = '';
} else {
card.style.display = 'none';
}
});
}
</script>

</html>
```

14. about.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>BeiRadar – Hot Deals</title>
<link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
```

```
<style>
/* Deals Hero Section */
.deals-hero {
background: white;
padding: 60px 20px;
text-align: center;
margin-bottom: 30px;
margin-top: 30px;
border-bottom: 1px solid #eee;
}

.deals-hero h1 {
font-size: 2.5rem;
font-weight: 700;
color: #111;
margin-bottom: 15px;
}

.deals-hero p {
font-size: 1rem;
color: #666;
margin-bottom: 20px;
}

.deals-count {
display: inline-block;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 10px 20px;
border-radius: 20px;
font-size: 0.9rem;
font-weight: 600;
}

/* Store Filter Section */
```

```
.store-filter {
display: flex;
justify-content: center;
gap: 12px;
flex-wrap: wrap;
margin-top: 20px;
}

.store-filter-btn {
background: #f5f5f5;
color: #333;
border: 2px solid #ddd;
padding: 10px 18px;
border-radius: 20px;
cursor: pointer;
font-weight: 600;
font-size: 0.9rem;
transition: all 0.2s;
}

.store-filter-btn:hover {
background: #efefef;
border-color: #667eea;
}

.store-filter-btn.active {
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border-color: #667eea;
}

/* Deals Grid */
.deals-grid {
display: grid;
grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
```

```
gap: 25px;
padding: 30px;
max-width: 1400px;
margin: 0 auto;
}

.deal-card {
background: white;
border-radius: 16px;
overflow: hidden;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
transition: transform 0.2s, box-shadow 0.2s;
display: flex;
flex-direction: column;
}

.deal-card:hover {
transform: translateY(-5px);
box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
}

/* Discount Badge */
.discount-badge {
position: absolute;
top: 15px;
right: 15px;
background: #ff6b6b;
color: white;
padding: 8px 12px;
border-radius: 8px;
font-weight: 700;
font-size: 0.85rem;
z-index: 10;
}
```

```css
/* Product Image */
.deal-image-container {
width: 100%;
height: 200px;
display: flex;
align-items: center;
justify-content: center;
background: #f9f9f9;
padding: 15px;
position: relative;
}

.deal-image {
max-width: 100%;
max-height: 100%;
object-fit: contain;
}

/* Deal Content */
.deal-content {
padding: 20px;
flex: 1;
display: flex;
flex-direction: column;
}

.deal-product-name {
font-size: 1rem;
font-weight: 700;
color: #111;
margin: 0 0 8px 0;
line-height: 1.3;
}

.deal-category {
```

```
color: #666;
font-size: 0.85rem;
margin-bottom: 10px;
}

.deal-store-badge {
display: inline-block;
background: #667eea;
color: white;
padding: 6px 12px;
border-radius: 6px;
font-size: 0.8rem;
font-weight: 600;
margin-bottom: 12px;
width: fit-content;
}

/* Pricing Section */
.deal-pricing {
background: #f9f9f9;
padding: 15px;
border-radius: 12px;
margin-top: auto;
margin-bottom: 15px;
}

.price-row {
display: flex;
justify-content: space-between;
align-items: center;
margin-bottom: 10px;
}

.price-row:last-child {
margin-bottom: 0;
```

```
}

.price-label {
color: #666;
font-size: 0.9rem;
font-weight: 600;
}

.original-price {
font-size: 1rem;
color: #999;
text-decoration: line-through;
}

.deal-price {
font-size: 1.4rem;
font-weight: 700;
color: #4caf50;
}

.savings {
font-size: 0.95rem;
color: #ff6b6b;
font-weight: 600;
padding-top: 10px;
border-top: 1px solid #eee;
margin-top: 10px;
}

/* Add to Cart Button */
.deal-add-btn {
width: 100%;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
border: none;
```

```
padding: 12px 15px;
border-radius: 10px;
font-size: 0.95rem;
font-weight: 600;
cursor: pointer;
transition: transform 0.2s;
}

.deal-add-btn:hover {
transform: scale(1.05);
}

/* No Deals */
.no-deals {
text-align: center;
padding: 80px 20px;
color: #666;
}

.no-deals h2 {
font-size: 1.5rem;
color: #111;
margin-bottom: 10px;
}

.no-deals p {
font-size: 1.05rem;
margin-bottom: 20px;
}

.no-deals a {
display: inline-block;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
color: white;
padding: 12px 30px;
```

```
border-radius: 8px;
text-decoration: none;
font-weight: 600;
}

/* Responsive */
@media (max-width: 768px) {
.deals-hero {
padding: 40px 20px;
}

.deals-hero h1 {
font-size: 1.8rem;
}

.deals-grid {
grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
gap: 15px;
padding: 15px;
}

.discount-badge {
padding: 8px 12px;
font-size: 0.85rem;
}
}
</style>
</head>

<body>
<header class="nav">
<div class="brand">
<span class="brand-icon">📡</span>
BeiRadar
</div>
```

```
<div class="hamburger" id="hamburger">
<span></span>
<span></span>
<span></span>
</div>

<nav class="nav-center" id="nav-links">
<a href="{{ url_for('home') }}">Home</a>
<a href="{{ url_for('categories_list') }}">Categories</a>
<a href="{{ url_for('deals') }}">Deals</a>
<a href="{{ url_for('about') }}">About</a>
</nav>

<div class="nav-right">
<a href="#" class="nav-notif">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M12 22c1.1 0 2-.9 2-2h-4c0 1.1.9 2 2 2zm6-6v-5c0-3.07-1.63-5.64-4.5-
6.32V4c0-.83-.67-1.5-1.5-1.5s-1.5.67-1.5 1.5v.68C7.64 5.36 6 7.92 6 11v5l-2
2v1h16v-1l-2-2zm-2 1H8v-6c0-2.48 1.51-4.5 4-4.5s4 2.02 4 4.5v6z"/>
</svg>
<span class="notif-badge">3</span>
</a>

<a href="{{ url_for('cart_view') }}" class="nav-cart">
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24">
<path d="M7 18c-1.1 0-1.99.9-1.99 2S5.9 22 7 22s2-.9 2-2-.9-2-2-2zM1 2v2h2l3.6
7.59-1.35 2.45c-.16.28-.25.61-.25.96 0 1.1.9 2 2 2h12v-2H7.42c-.14 0-.25-.11-.25-
.25l.03-.12.9-1.63h7.45c.75 0 1.41-.41 1.75-1.03l3.58-6.49c.08-.14.12-.31.12-.48
0-.55-.45-1-1-1H5.21l-.94-2H1zm16 16c-1.1 0-1.99.9-1.99 2s.89 2 1.99 2 2-.9 2-2-
.9-2-2-2z"/>
</svg>
{% if session.get('cart') %}
<span class="cart-badge">{{ session['cart']|length }}</span>
{% endif %}
```

```
</a>
</div>
</header>

<!-- Deals Hero Section -->
<section class="deals-hero">
<h1>🔥 Hot Deals</h1>
<p>Get the biggest discounts across Nairobi's top supermarkets</p>
{% if deals %}
<div class="deals-count">{{ deals|length }} Amazing Deals Available</div>
{% endif %}

<!-- Store Filter -->
<div class="store-filter">
<button class="store-filter-btn active" onclick="filterDeals('all')">All
Stores</button>
<button class="store-filter-btn" onclick="filterDeals('Carrefour')">🏪
Carrefour</button>
<button class="store-filter-btn" onclick="filterDeals('Naivas')">🏪
Naivas</button>
<button class="store-filter-btn" onclick="filterDeals('Quickmart')">🏪
Quickmart</button>
</div>
</section>

<!-- Deals Grid Section -->
<section>
{% if deals %}
<div class="deals-grid">
{% for deal in deals %}
<div class="deal-card" data-store="{{ deal.store }}">
<!-- Discount Badge -->
<div class="discount-badge">
🎉 {{ "{:.0f}".format(deal.deal_percentage) }}% OFF
```

```
</div>

<!-- Product Image -->
<div class="deal-image-container">
{% if deal.image_url %}
<img  src="{{  get_image_url(deal.image_url)  }}"  alt="{{  deal.product_name  }}"
class="deal-image">
{% else %}
<div style="color: #ccc; font-size: 3rem;">🛍️</div>
{% endif %}
</div>

<!-- Deal Content -->
<div class="deal-content">
<h3 class="deal-product-name">{{ deal.product_name }}</h3>
<p class="deal-category">{{ deal.category }}</p>

<!-- Store Badge -->
<div class="deal-store-badge">
🏪 {{ deal.store }}
</div>

<!-- Pricing -->
<div class="deal-pricing">
<div class="price-row">
<span class="price-label">Original Price:</span>
<span class="original-price">KSh {{ "{:,.0f}".format(deal.old_price) }}</span>
</div>
<div class="price-row">
<span class="price-label">Deal Price:</span>
<span class="deal-price">KSh {{ "{:,.0f}".format(deal.new_price) }}</span>
</div>
<div class="savings">
💰 Save KSh {{ "{:,.0f}".format(deal.old_price - deal.new_price) }}
</div>
```

```
</div>

<!-- Add to Cart Button -->
<form method="POST" action="{{ url_for('cart_add', product_name=deal.product_name)
}}">
<input type="hidden" name="quantity" value="1">
<button type="submit" class="deal-add-btn">🛒 Add Deal to Cart</button>
</form>
</div>
</div>
{% endfor %}
</div>
{% else %}
<div class="no-deals">
<h2>No Hot Deals Right Now</h2>
<p>Check back later for amazing discounts!</p>
<a href="{{ url_for('home') }}">← Browse All Products</a>
</div>
{% endif %}
</section>

<footer>
<p>© 2025 BeiRadar</p>
</footer>

</body>

<script>
const hamburger = document.getElementById('hamburger');
const navLinks = document.getElementById('nav-links');

hamburger.addEventListener('click', () => {
navLinks.classList.toggle('active');
hamburger.classList.toggle('open');
});
```

```
function filterDeals(store) {
const dealCards = document.querySelectorAll('.deal-card');
const filterBtns = document.querySelectorAll('.store-filter-btn');

// Update active button
filterBtns.forEach(btn => btn.classList.remove('active'));
event.target.classList.add('active');

// Filter deals
dealCards.forEach(card => {
if (store === 'all' || card.dataset.store === store) {
card.style.display = '';
} else {
card.style.display = 'none';
}
});
}
</script>

</html>
```

15. 404.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Page Not Found</title>
</head>
<body>
<h1>404 - Page Not Found</h1>
<p>Sorry, the page you are looking for does not exist.</p>
<a href="{{ url_for('home') }}">Go Home</a>
```

```
</body>
</html>
```

15. styles.css

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&di
splay=swap');

/* ---------------------- Base Styles ---------------------- */
* {
margin: 0;
padding: 0;
box-sizing: border-box;
font-family: "Poppins", sans-serif;
}

body {
background: #f4f5fa;
overflow-x: hidden;
position: relative;
}

.product-image {
width: 180px;
height: 180px;
object-fit: contain;
margin-bottom: 10px;
}


/* ---------------------- Decorative Circles ---------------------- */
.circle {
position: absolute;
border-radius: 50%;
```

```
filter: blur(60px);
opacity: 0.5;
z-index: -1;
}

.c1 {
width: 350px;
height: 350px;
background: #c6d2ff;
top: -100px;
right: -100px;
}

.c2 {
width: 300px;
height: 300px;
background: #ffe5f0;
bottom: -100px;
left: -100px;
}
/* ---------------------- Navbar ---------------------- */
.nav {
display: flex;
justify-content: space-between;
align-items: center;
padding: 0 50px;
height: 80px;
background: rgba(255, 255, 255, 0.95);
backdrop-filter: blur(20px);
box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
position: fixed;
top: 0;
left: 0;
right: 0;
z-index: 1000;
```

```
}

/* Brand Logo */
.brand {
display: flex;
align-items: center;
gap: 10px;
font-size: 32px;
font-weight: 800;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
-webkit-background-clip: text;
-webkit-text-fill-color: transparent;
background-clip: text;
cursor: pointer;
transition: transform 0.3s ease;
}

.brand:hover {
transform: scale(1.05);
}

.brand-icon {
font-size: 36px;
background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
-webkit-background-clip: text;
-webkit-text-fill-color: transparent;
animation: pulse 2s ease-in-out infinite;
}

@keyframes pulse {
0%, 100% { transform: scale(1); }
50% { transform: scale(1.1); }
}

/* Navigation Center */
```

```css
.nav-center {
display: flex;
align-items: center;
gap: 35px;
}

.nav-center a {
text-decoration: none;
color: #444;
font-weight: 600;
font-size: 16px;
padding: 10px 20px;
border-radius: 12px;
position: relative;
transition: all 0.3s cubic-bezier(0.4, 0, 0.2, 1);
overflow: hidden;
}

.nav-center a::before {
content: '';
position: absolute;
top: 0;
left: -100%;
width: 100%;
height: 100%;
background: linear-gradient(135deg, #667eea, #764ba2);
transition: left 0.3s ease;
z-index: -1;
border-radius: 12px;
}

.nav-center a:hover::before,
.nav-center a.active::before {
left: 0;
}
```

```
.nav-center a:hover,
.nav-center a.active {
color: white;
transform: translateY(-3px);
box-shadow: 0 8px 20px rgba(102, 126, 234, 0.4);
}

/* Right Side Icons */
.nav-right {
display: flex;
align-items: center;
gap: 25px;
}

/* Notification Icon */
.nav-notif {
position: relative;
cursor: pointer;
transition: all 0.3s ease;
}

.nav-notif svg {
width: 28px;
height: 28px;
fill: #444;
transition: all 0.3s ease;
}

.nav-notif:hover {
transform: scale(1.15) rotate(15deg);
}

.nav-notif:hover svg {
fill: #667eea;
```

```
filter: drop-shadow(0 0 10px rgba(102, 126, 234, 0.5));
}

.notif-badge {
position: absolute;
top: -8px;
right: -8px;
background: linear-gradient(135deg, #ff6b6b, #ff4757);
color: white;
font-size: 11px;
font-weight: 700;
padding: 3px 7px;
border-radius: 50%;
box-shadow: 0 4px 12px rgba(255, 75, 92, 0.4);
animation: bounce 2s ease-in-out infinite;
}

@keyframes bounce {
0%, 100% { transform: scale(1); }
50% { transform: scale(1.2); }
}

/* Shopping Cart */
.nav-cart {
position: relative;
cursor: pointer;
transition: all 0.3s ease;
}

.nav-cart svg {
width: 32px;
height: 32px;
stroke: #444;
fill: none;
stroke-width: 2;
```

```
transition: all 0.3s ease;
}

.nav-cart:hover {
transform: scale(1.2);
}

.nav-cart:hover svg {
stroke: #667eea;
filter: drop-shadow(0 0 15px rgba(102, 126, 234, 0.6));
}

.cart-badge {
position: absolute;
top: -10px;
right: -10px;
background: linear-gradient(135deg, #667eea, #764ba2);
color: white;
font-size: 12px;
font-weight: 700;
padding: 4px 8px;
border-radius: 50%;
box-shadow: 0 4px 12px rgba(102, 126, 234, 0.4);
min-width: 22px;
text-align: center;
}

/* Add Cart Button */
.add-cart-btn {
padding: 10px 20px;
background: linear-gradient(135deg, #667eea, #764ba2);
color: #fff;
border: none;
border-radius: 10px;
cursor: pointer;
```

```
font-weight: 600;
transition: all 0.3s ease;
box-shadow: 0 4px 15px rgba(102, 126, 234, 0.3);
}

.add-cart-btn:hover {
transform: translateY(-2px);
box-shadow: 0 6px 20px rgba(102, 126, 234, 0.5);
background: linear-gradient(135deg, #764ba2, #667eea);
}


/* Smooth scroll */
html {
scroll-behavior: smooth;
}

/* Hamburger Menu Styles */
.hamburger {
display: none;
flex-direction: column;
justify-content: space-between;
width: 28px;
height: 22px;
cursor: pointer;
position: relative;
z-index: 3000; /* stays above everything */
}

.hamburger span {
display: block;
height: 3px;
width: 100%;
background: #111;
border-radius: 2px;
```

```
transition: all 0.3s;
}

/* Mobile Navigation */
@media (max-width: 768px) {

.nav {
display: flex;
align-items: center;
justify-content: space-between;
padding: 20px 25px;
height: 75px;
}

.hamburger {
display: flex !important;
margin-left: 20px;
transform: scale(1.1);
z-index: 3000;
}

.brand {
margin-right: auto;
font-size: 26px;
}

.nav-center {
position: absolute;
top: 75px;
left: 0;
right: 0;
background: white;
flex-direction: column;
align-items: center;
gap: 25px;
```

```
padding: 30px 0;
display: none;
z-index: 1000;
box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
}

.nav-center.active {
display: flex;
}
}

/* ----------------------- Hero Section ----------------------- */
.hero {
text-align: center;
margin-top: 80px;
}

.hero h1 {
font-size: 50px;
font-weight: 700;
color: #1f2a56;
}

.tagline {
font-size: 18px;
margin-top: 10px;
color: #666;
}

/* Search Box */
.search-box {
display: flex;
align-items: center;
width: 90%;
max-width: 700px;
```

```
margin: 40px auto;
background: white;
padding: 10px 15px;
border-radius: 60px;
box-shadow: 0 6px 35px rgba(0,0,0,0.1);
position: relative;
}

.search-box input {
flex: 1;
padding: 12px 20px;
border: none;
outline: none;
border-radius: 40px;
font-size: 16px;
}

.search-box button {
flex-shrink: 0;
padding: 12px 25px;
margin-left: 10px;
border: none;
border-radius: 50px;
background: linear-gradient(135deg, #6b7cff, #a98eff);
color: white;
font-weight: 600;
cursor: pointer;
transition: all 0.3s ease;
}

.search-box button:hover {
transform: translateY(-2px);
box-shadow: 0 6px 15px rgba(107,124,255,0.4);
}
```

```css
/* Suggestions Dropdown */
.suggestions {
position: absolute;
top: 100%;
left: 0;
width: 100%;
background: white;
border-radius: 12px;
box-shadow: 0 8px 25px rgba(0, 0, 0, 0.15);
list-style: none;
padding: 0;
margin-top: 8px;
z-index: 1000;
max-height: 250px;
overflow-y: auto;
display: none;
}

.suggestions li {
padding: 12px 18px;
cursor: pointer;
font-size: 14px;
color: #1f2a56;
transition: all 0.2s ease;
}

.suggestions li:hover {
background: linear-gradient(135deg, #6b7cff, #a98eff);
color: white;
border-radius: 8px;
}

.suggestions li .match {
font-weight: 600;
color: #6b7cff;
```

```
}

.suggestions::-webkit-scrollbar {
width: 6px;
}

.suggestions::-webkit-scrollbar-thumb {
background: #6b7cff;
border-radius: 3px;
}

.suggestions::-webkit-scrollbar-track {
background: #f4f5fa;
}

.result {
width: 95%;
margin: 40px auto 80px auto;
}

.product-card {
background: #fff;
border-radius: 20px;
padding: 20px 30px;
margin-bottom: 30px;
box-shadow: 0 12px 40px rgba(0,0,0,0.08);
transition: all 0.3s ease;
display: flex;
flex-direction: column;
width: 100%;
}

.product-card:hover {
transform: translateY(-4px);
box-shadow: 0 16px 50px rgba(0,0,0,0.1);
```

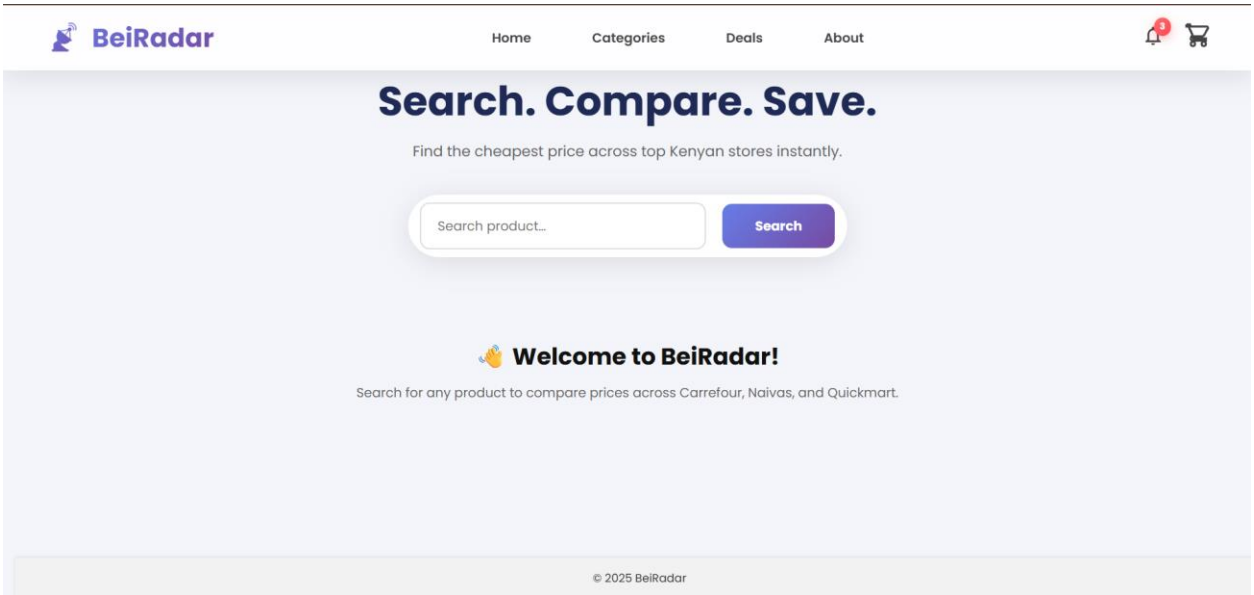
```
}

/* Top strip: Best Price / Typical Price */
.product-card .top-strip {
display: flex;
justify-content: space-between;
align-items: center;
padding: 12px 20px;
border-radius: 12px;
background: linear-gradient(135deg, #6b7cff, #a98eff);
color: white;
font-weight: 600;
margin-bottom: 15px;
}

/* Optional “On Sale” tag */
.product-card .sale-tag {
padding: 4px 10px;
background: #ff6b6b;
color: white;
border-radius: 14px;
font-size: 12px;
font-weight: 600;
}

/* Stores container */
.compare {
display: flex;
gap: 20px;
justify-content: flex-start;
flex-wrap: wrap;
}

/* Each store box */
.store {
```

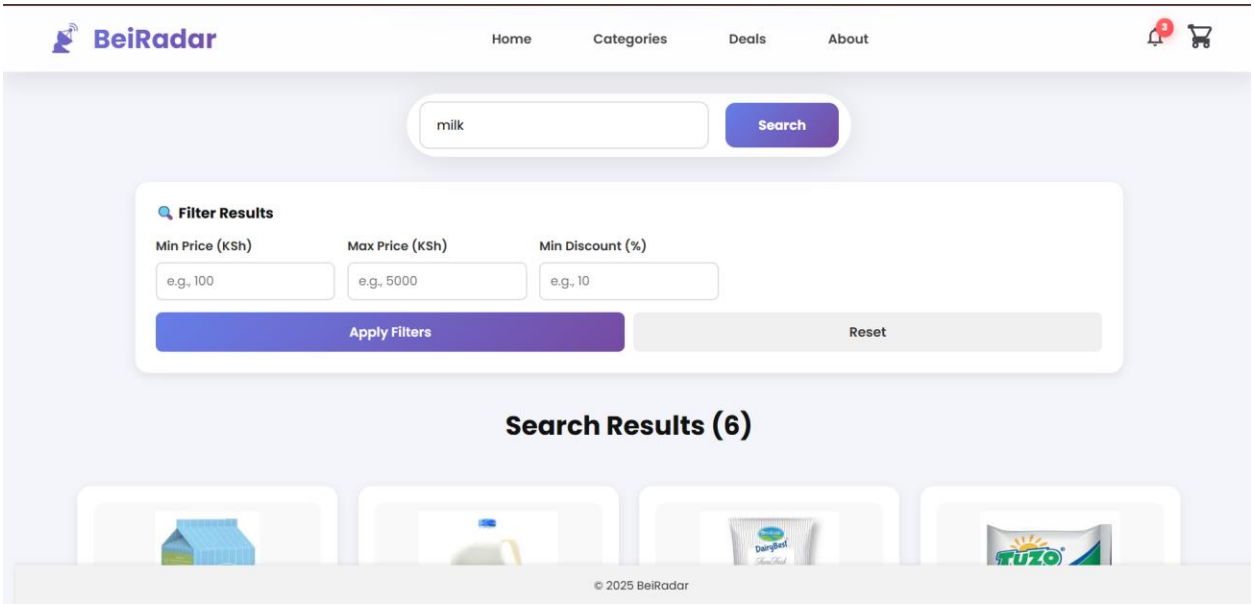
```
flex: 1 1 150px;
background: #f7f8fc;
padding: 15px 20px;
border-radius: 15px;
text-align: center;
transition: all 0.3s ease;
position: relative;
}

.store:hover {
transform: translateY(-3px);
box-shadow: 0 10px 30px rgba(0,0,0,0.08);
}

.store h4 {
font-size: 16px;
font-weight: 600;
color: #1f2a56;
margin-bottom: 8px;
}

.store .price {
font-size: 20px;
font-weight: 700;
}

/* Best Deal highlight */
.s-best {
border: 2px solid #6b7cff;
background: #e4e7ff;
}

.store .deal-tag {
position: absolute;
top: -10px;
```

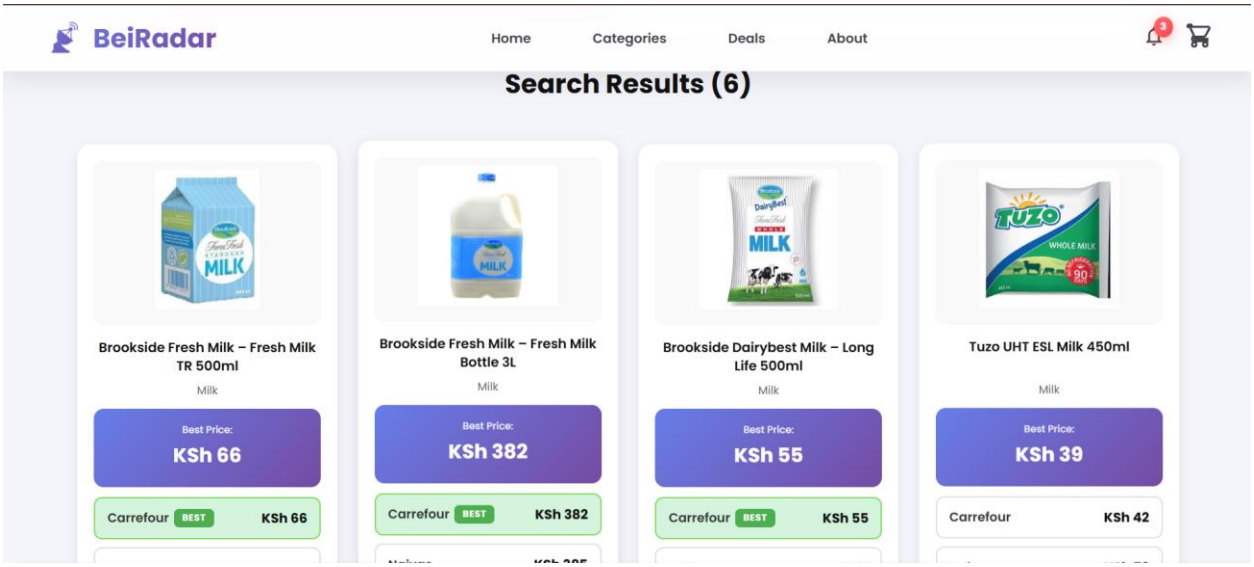
```
right: -10px;
background: #1e90ff;
color: white;
font-size: 12px;
font-weight: 600;
padding: 3px 8px;
border-radius: 12px;
}

/* ----------------------- Global Responsive ----------------------- */
html {
font-size: 16px;
}

body {
padding: 0 15px;
}

/* Images scale nicely */
img {
max-width: 100%;
height: auto;
}

/* Make all containers flexible */
.container {
width: 100%;
max-width: 1200px;
margin: 0 auto;
padding: 0 15px;
}

/* Flex helper classes */
.flex {
display: flex;
```

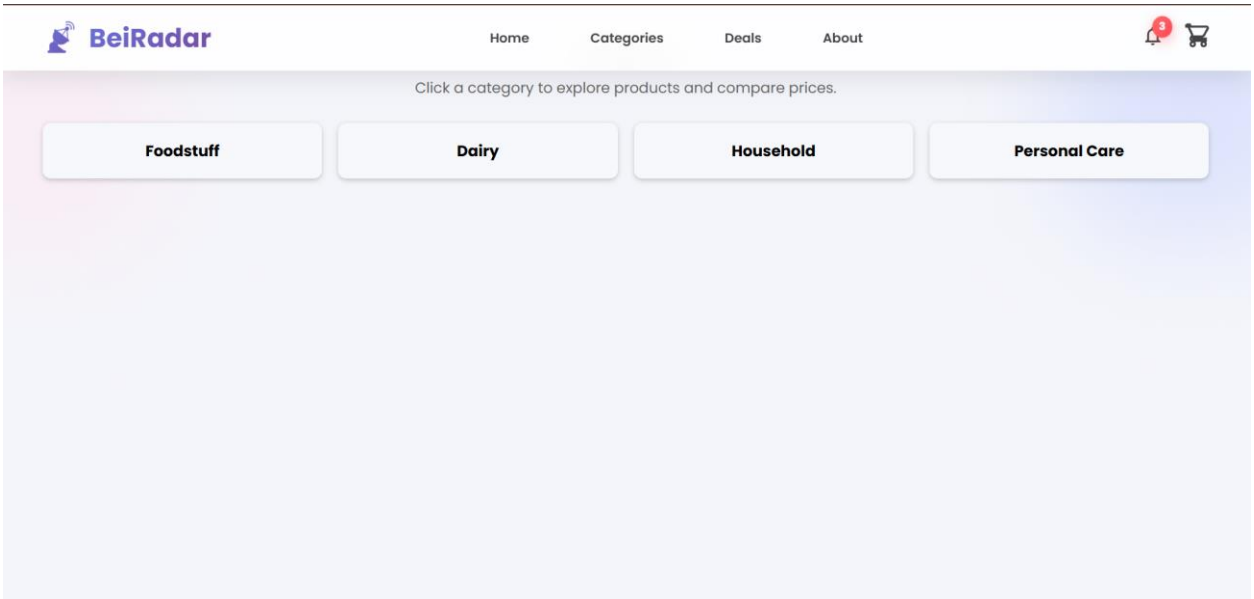
```
flex-wrap: wrap;
gap: 20px;
}

.flex-center {
justify-content: center;
align-items: center;
}

footer {
background-color: #f2f2f2;
color: #333;
text-align: center;
padding: 15px 0;
position: fixed;
bottom: 0;
width: 100%;
font-size: 14px;
box-shadow: 0 -1px 5px rgba(0, 0, 0, 0.1);
```

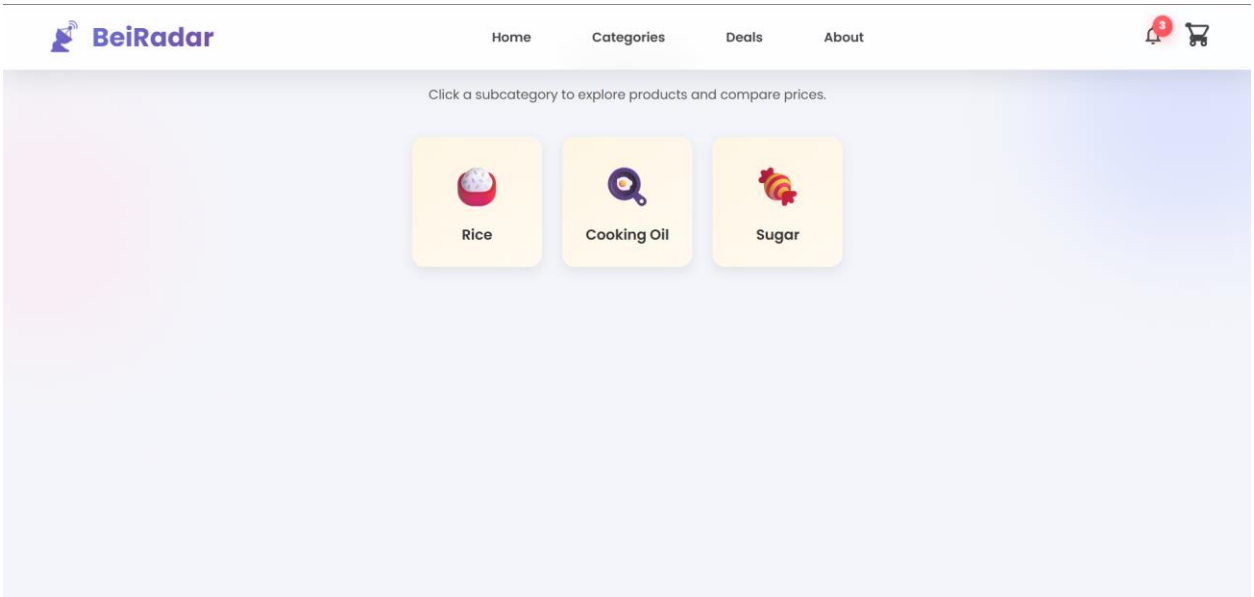
5.2 Screenshots

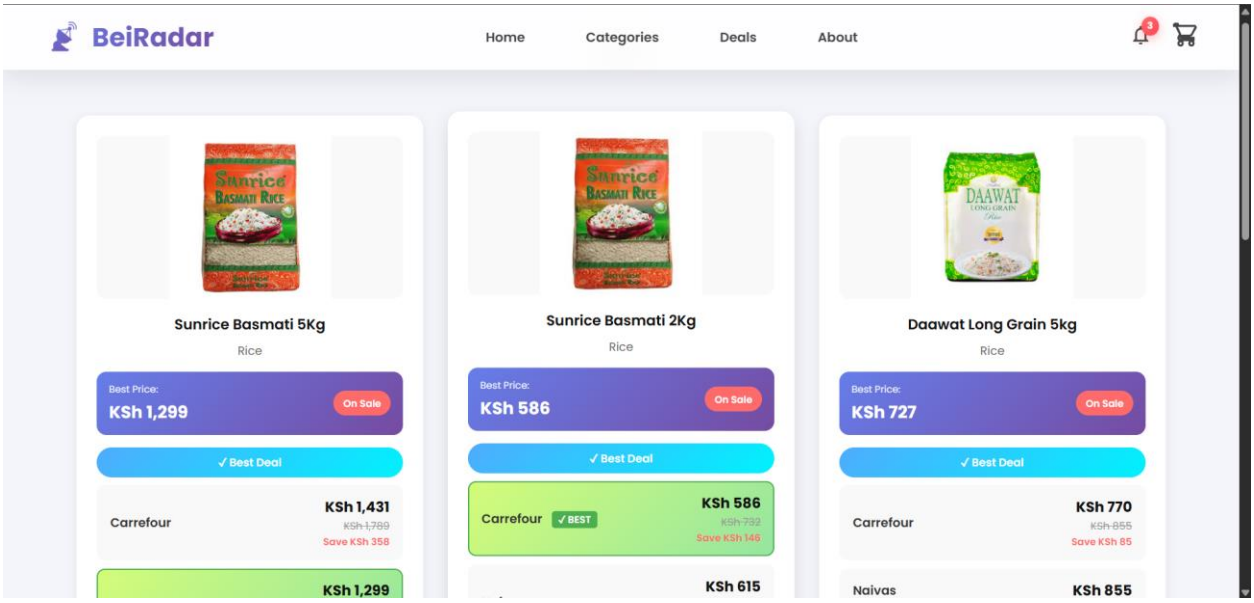


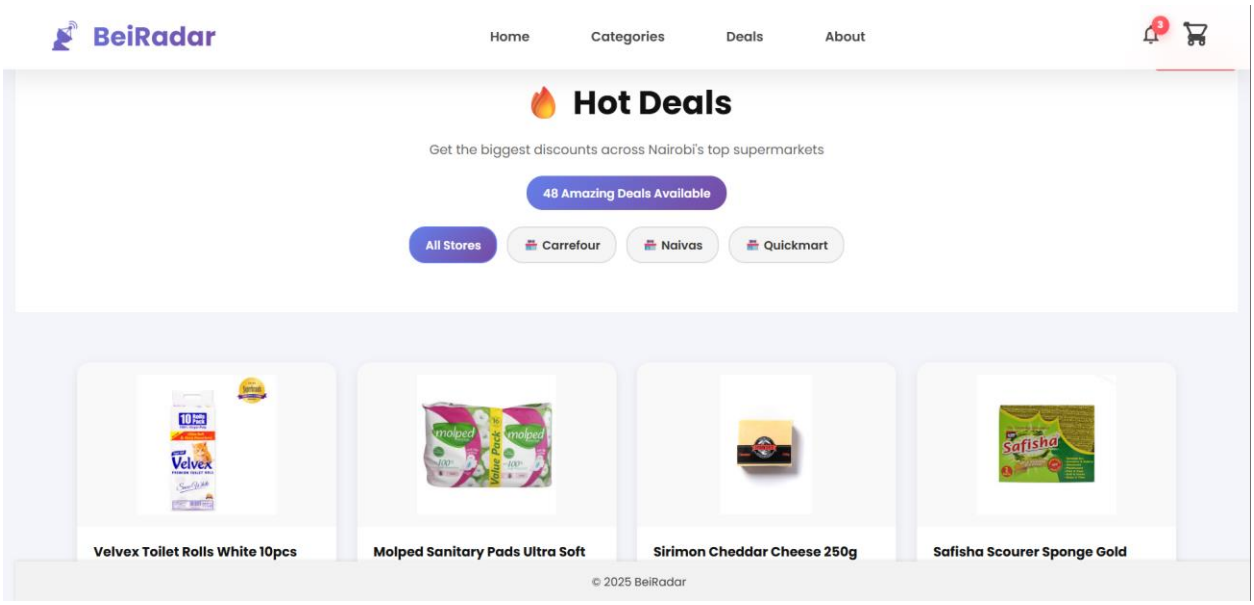

BeiRadar
Home
Categories
Deals
About
milk
Search
Filter Results
Min Price (KSh)
Max Price (KSh)
Min Discount (%)
e.g., 100
e.g., 5000
e.g., 10
Apply Filters
Reset
Search Results (6)
© 2025 BeiRadar

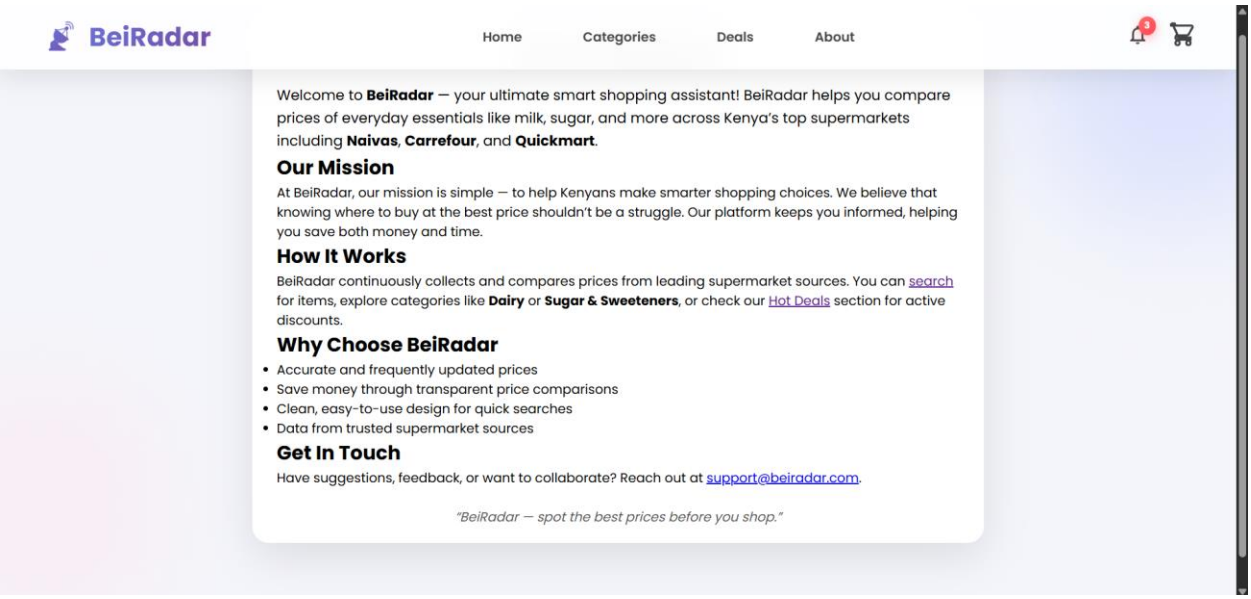

BeiRadar
Home
Categories
Deals
About
Search Results (6)
Brookside Fresh Milk – Fresh Milk TR 500ml
Milk
Best Price:
KSh 66
Carrefour BEST KSh 66
Brookside Fresh Milk – Fresh Milk Bottle 3L
Milk
Best Price:
KSh 382
Carrefour BEST KSh 382
Brookside Dairybest Milk – Long Life 500ml
Milk
Best Price:
KSh 55
Carrefour BEST KSh 55
Tuzo UHT ESL Milk 450ml
Milk
Best Price:
KSh 39
Carrefour KSh 42

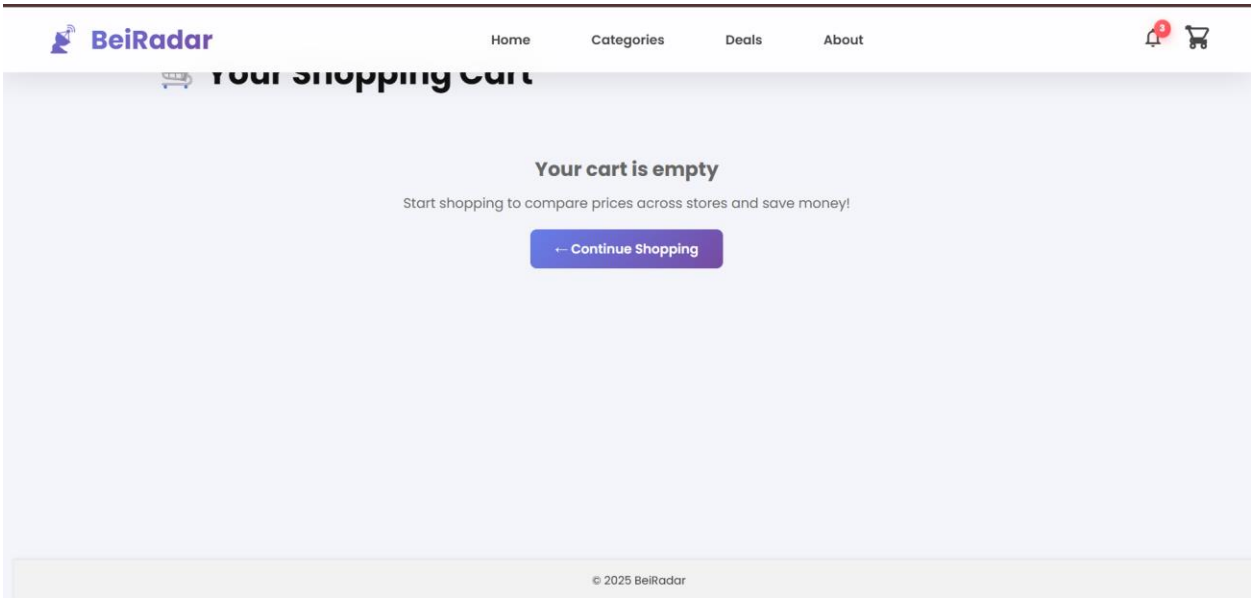

BeiRadar
Home
Categories
Deals
About
3
Click a category to explore products and compare prices.
Foodstuff
Dairy
Household
Personal Care

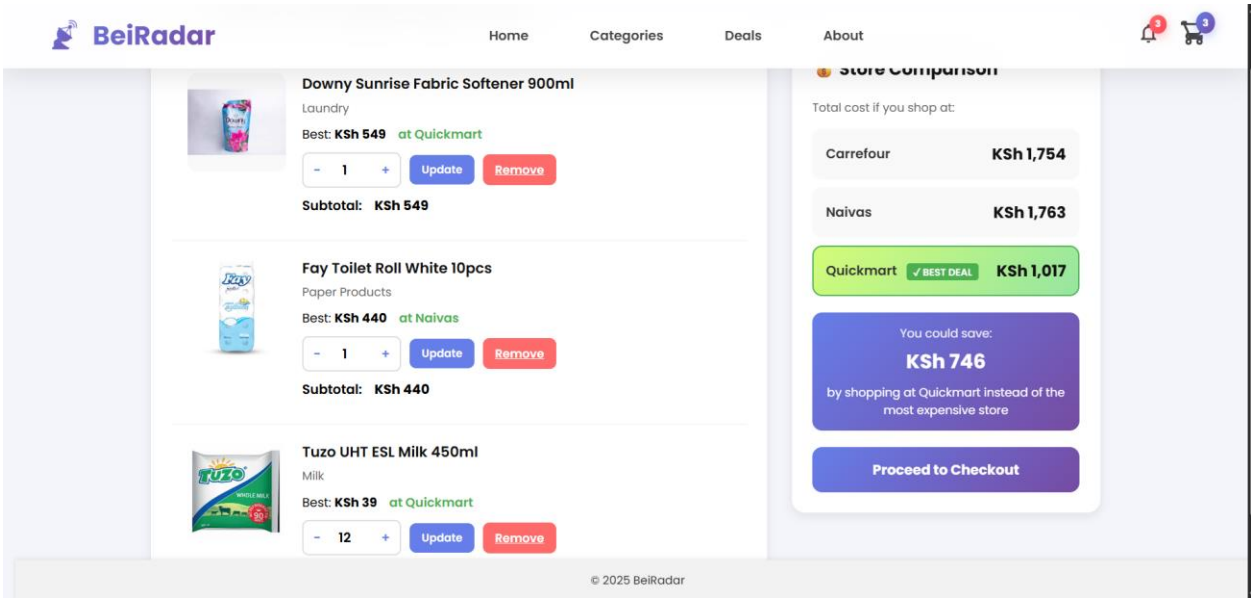

BeiRadar
Home
Categories
Deals
About
3
Click a subcategory to explore products and compare prices.
Rice
Cooking Oil
Sugar


BeiRadar
Home
Categories
Deals
About
Sunrice Basmati 5Kg
Rice
Best Price:
KSh 1,299
On Sale
✓ Best Deal
Carrefour
KSh 1,431
KSh 1,789
Save KSh 358
KSh 1,299
Sunrice Basmati 2Kg
Rice
Best Price:
KSh 586
On Sale
✓ Best Deal
Carrefour
✓ BEST
KSh 586
KSh 732
Save KSh 146
KSh 615
Daawat Long Grain 5kg
Rice
Best Price:
KSh 727
On Sale
✓ Best Deal
Carrefour
KSh 770
KSh 855
Save KSh 85
Naivas
KSh 855


BeiRadar
Home
Categories
Deals
About
Hot Deals
Get the biggest discounts across Nairobi's top supermarkets
48 Amazing Deals Available
All Stores
Carrefour
Naivas
Quickmart
Velvex Toilet Rolls White 10pcs
Molped Sanitary Pads Ultra Soft
Sirimon Cheddar Cheese 250g
Safisha Scourer Sponge Gold
© 2025 BeiRadar


BeiRadar
Home
Categories
Deals
About
Welcome to BeiRadar — your ultimate smart shopping assistant! BeiRadar helps you compare prices of everyday essentials like milk, sugar, and more across Kenya's top supermarkets including Naivas, Carrefour, and Quickmart.
Our Mission
At BeiRadar, our mission is simple — to help Kenyans make smarter shopping choices. We believe that knowing where to buy at the best price shouldn't be a struggle. Our platform keeps you informed, helping you save both money and time.
How It Works
BeiRadar continuously collects and compares prices from leading supermarket sources. You can search for items, explore categories like Dairy or Sugar & Sweeteners, or check our Hot Deals section for active discounts.
Why Choose BeiRadar
Accurate and frequently updated prices
Save money through transparent price comparisons
Clean, easy-to-use design for quick searches
Data from trusted supermarket sources
Get In Touch
Have suggestions, feedback, or want to collaborate? Reach out at support@beiradar.com.
"BeiRadar — spot the best prices before you shop."


BeiRadar
Home
Categories
Deals
About
Your cart is empty
Start shopping to compare prices across stores and save money!
← Continue Shopping
© 2025 BeiRadar


BeiRadar
Home
Categories
Deals
About
Downy Sunrise Fabric Softener 900ml
Laundry
Best: KSh 549 at Quickmart
- 1 + Update Remove
Subtotal: KSh 549
Fay Toilet Roll White 10pcs
Paper Products
Best: KSh 440 at Naivas
- 1 + Update Remove
Subtotal: KSh 440
Tuzo UHT ESL Milk 450ml
Milk
Best: KSh 39 at Quickmart
- 12 + Update Remove
Total cost if you shop at:
Carrefour KSh 1,754
Naivas KSh 1,763
Quickmart ✓ BEST DEAL KSh 1,017
You could save:
KSh 746
by shopping at Quickmart instead of the most expensive store
Proceed to Checkout
© 2025 BeiRadar

CHAPTER 6

6.0 Conclusions

The BeiRadar project has successfully developed a web-based platform that enables users to compare product prices across major supermarkets in Kenya, including Carrefour, Naivas, and Quickmart. The system integrates a centralized SQLite database and Python-based backend scripts, ensuring accurate and up-to-date product information.

The backend, implemented with Flask, efficiently handles product queries, filtering, and cart management. Additionally, the image-matching script enhances the user experience by linking products to their respective images, minimizing manual effort and ensuring clarity. The platform's price calculation algorithms provide users with insights into the most cost-effective store for each product, allowing for informed shopping decisions.

Overall, BeiRadar demonstrates the effective use of database management, backend development, and data processing techniques. The project also highlights the challenges of inconsistent product names and image mapping, which were successfully mitigated using a combination of text normalization, fuzzy matching algorithms, and manual mappings.

6.2 Recommendations

To further enhance the functionality and usability of BeiRadar, the following recommendations are proposed:

1. **Automated Data Updates:** Integrate supermarket APIs to automatically fetch product prices, reducing manual Excel uploads and ensuring real-time accuracy.
2. **Expanded Product Categories:** Include additional supermarkets and product categories to increase the platform's comprehensiveness and relevance to a broader user base.
3. **User Personalization:** Introduce user accounts that track shopping preferences, enabling personalized product recommendations and notifications for special deals.

4. **Enhanced User Interface:** Improve the frontend design to facilitate easier navigation, advanced filtering, and mobile responsiveness.
5. **Mobile Application Development:** Create a mobile version of BeiRadar to cater to users who prefer shopping on smartphones and tablets, increasing accessibility.
6. **Data Analytics and Insights:** Implement features for trend analysis, such as price changes over time or identification of highly discounted products, providing users with actionable insights.

These enhancements would improve the scalability, usability, and overall impact of the platform.

6.3 References

1. Kenya National Bureau of Statistics. (2025, August 29). *Consumer price indices and inflation rates – August 2025*. https://www.knbs.or.ke/wp-content/uploads/2025/08/Kenya-Consumer-Price-Indices-and-Inflation-Rates-August-2025.pdf
2. Reuters. (2025, August 29). *Kenya's inflation rises in August, driven by food, transport prices*. https://www.reuters.com/world/africa/kenyas-inflation-rises-45-year-on-year-august-statistics-office-says-2025-08-29

6.4 Appendices

Appendix A: Backend Python Scripts

- Flask application for product management, cart functionality, and filtering.
- Database population script for importing Excel data into SQLite.
- Image-matching and assignment script for linking products with images.

Appendix B: Database Schema

- Table structure for `products` including sample records.
- Categories table and mappings (if applicable).

Appendix C: Screenshots

- Homepage displaying product search functionality.
- Category and subcategory pages.
- Cart view with multi-store comparison.
- Deals and discounts page.

Appendix D: Sample Outputs

- Product listings with price comparison results.
- Database diagnostic outputs.
- Unmatched product reports generated during image matching.