Analytics Summary

Firebase-Based Recipe Analytics Pipeline

1. Objective

This document summarizes the analytical outcomes of a Firebase-based recipe engagement platform built as part of the Data Engineering assessment. The goal of the analytics layer is to transform raw Firestore event data (recipes, users, and interactions) into actionable insights about recipe performance, user behavior, and content strategy using **only validated, clean data** produced by the upstream ETL and validation stages.

The primary dataset is seeded around the candidate's own recipe, **Idli Sambar**, complemented by additional synthetic recipes, users, and interactions to simulate a realistic consumer application.

2. Data & Methodology

2.1 Source Data

All analytics are computed on top of Firestore collections that model:

- **Recipes:** Idli Sambar + 18 synthetic recipes with realistic ingredients, steps, tags, and cuisines.
- **Users:** 30 users with location attributes (city, state, country).
- **Interactions:** 400 interaction events, covering views, likes, cook attempts, and ratings.
- **Injected "bad" data:** A small set of intentionally invalid records (e.g., negative prep time, invalid difficulty, out-of-range rating, invalid interaction type) used to test validation robustness.

These collections are seeded using firestore_setup.py as the first step of the pipeline.

2.2 ETL & Validation

The analytics layer never reads directly from Firestore. Instead, it consumes CSV outputs produced by upstream ETL and cleansed by a dedicated validation step:

1. **ETL (etl_export_transform.py):**
    - Streams Firestore collections (recipes, users, interactions).
    - Normalizes nested documents into relational tables:
        - recipe.csv
        - ingredients.csv

- steps.csv
- users.csv
- interactions.csv

2. **Validation (validation.py):**
    - Applies rule-based checks on required fields, numeric ranges, domain constraints (e.g., difficulty ∈ {Easy, Medium, Hard}), structural rules for steps/ingredients, and referential integrity for user/recipe links.
    - Separates records into:
        - clean_*.csv – used for analytics
        - quarantined_*.csv – invalid rows with reasons
    - Produces a consolidated validation_report.json under Validation_Output/.

Only the **clean tables** are loaded by analytics.py from Validation_Output/, ensuring all insights are based on consistent and trustworthy data.

2.3 Analytics Engine:

The analytics module (analytics.py) is the final stage in run_pipeline.py and performs the following:

- Loads clean_recipe.csv, clean_interactions.csv, clean_ingredients.csv, clean_users.csv, and clean_steps.csv.
- Computes **15 business and behavioral insights**.
- Generates:
    - analytics_report.json – machine-readable summary of all insights.
    - Supporting summary tables (e.g., interaction breakdowns).
    - Visual charts under Analytics_Output/Charts/ (bar, pie, histogram, and scatter plots).

The execution is fully orchestrated by run_pipeline.py, which runs:

Firestore seeding → 2. ETL → 3. Validation → 4. Analytics, with centralized logging provided by utils.py.

3. Metrics & Definitions

Across the analytics layer, a few core concepts are used repeatedly:

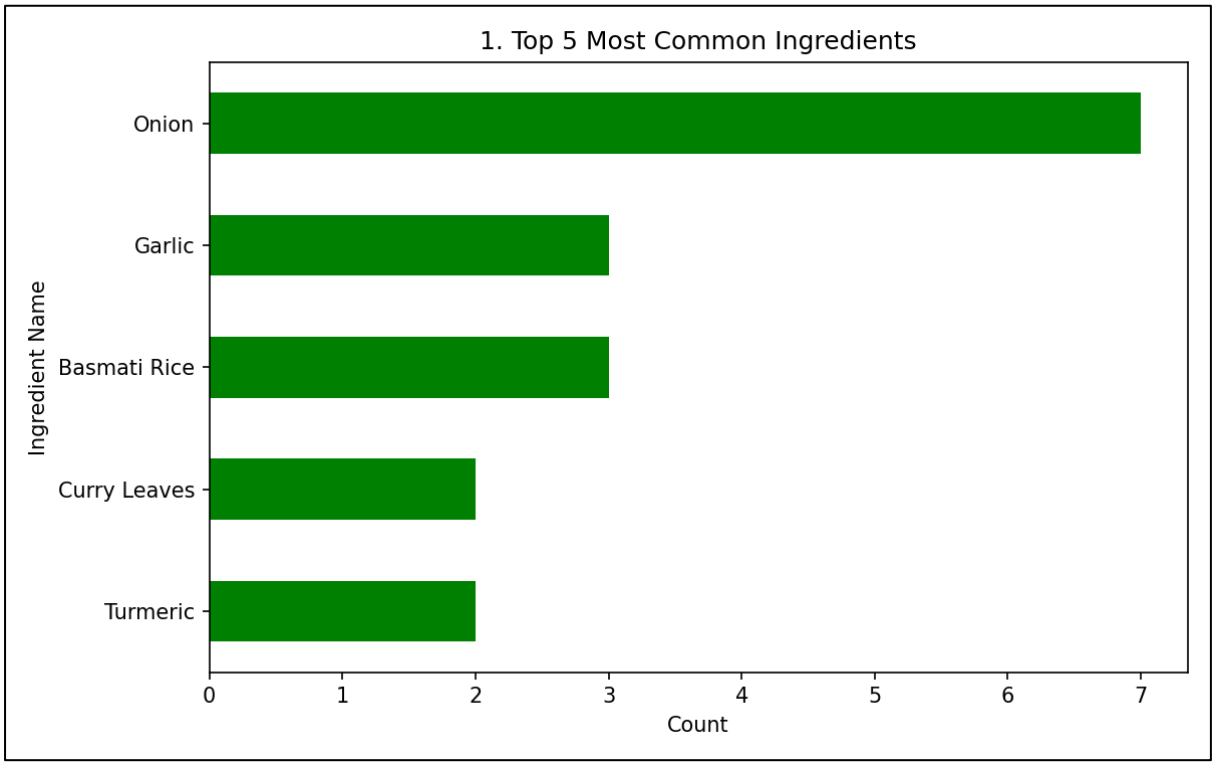
- **View:** A user opens a recipe (interaction type = "view").
- **Like:** A user explicitly likes a recipe (type = "like").
- **Cook Attempt:** A user indicates that they tried to cook a recipe (type = "cook_attempt").
- **Rating:** A 1 to 5 rating given for a recipe (type = "rating").
- **Engagement Score:** Count of non-view interactions (likes, cook attempts, ratings).
- **Weighted Popularity Score:** Composite score per recipe using different weights for each interaction type:
  - view = 1
  - like = 5
  - cook_attempt = 10
  - rating = 2

These metrics provide multiple lenses to understand recipe performance: volume (views), depth of interest (likes and cook attempts), and satisfaction (ratings).

4. Summary of Insights

Below is a structured overview of the 15 insights computed by analytics.py, with a focus on what was analyzed, how it was derived, and why it matters for the platform.

4.1 Most Common Ingredients



What: Identifies the top 5 most frequently used ingredients across all recipes using clean_ingredients.csv.

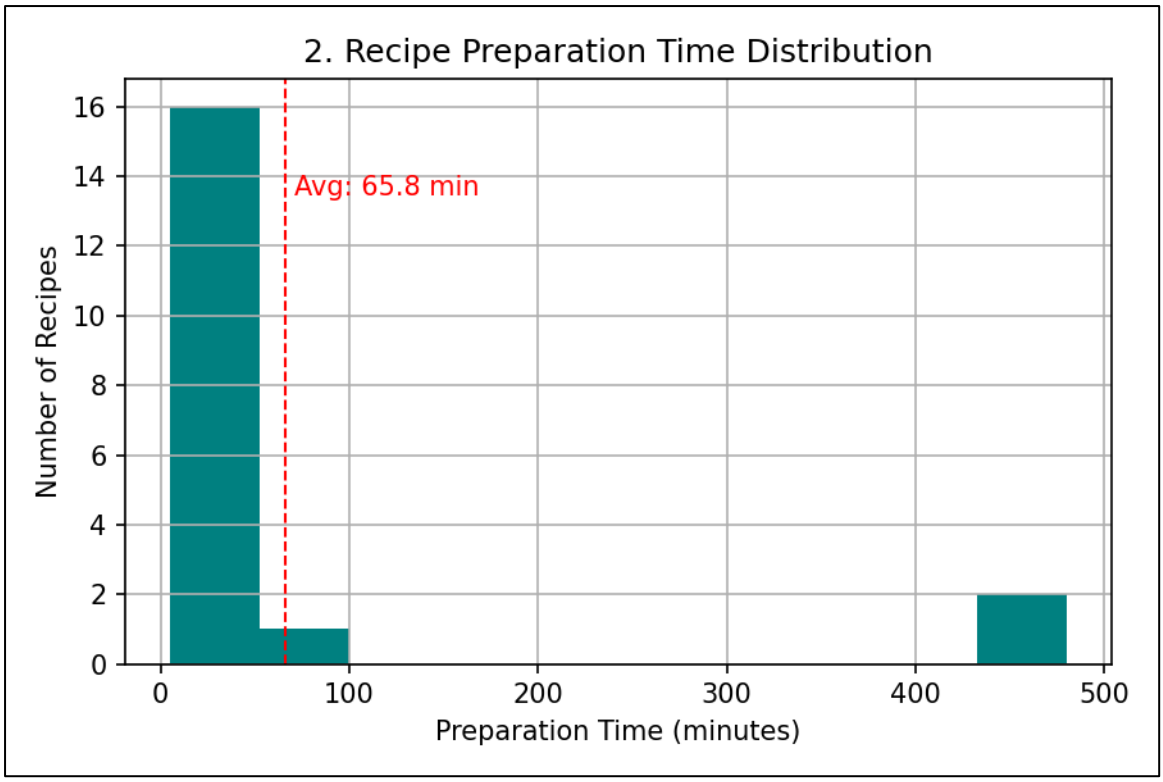
How: Counts ingredient names and ranks them by frequency; produces a bar chart "Top 5 Most Common Ingredients".

Key Observation (Qualitative): The dataset is dominated by staple Indian kitchen items (e.g., rice, onions, lentils, tomatoes, basic spices), indicating that the catalogue is centered around **everyday home cooking** rather than niche or gourmet dishes.

Why it matters:

- Confirms that ingredient choices are aligned with typical household availability.
- Supports ideas for inventory-based personalization (e.g., recommending recipes based on common pantry items).

4.2 Average Preparation Time



What: Measures the distribution and average of prep_time_minutes across all recipes.

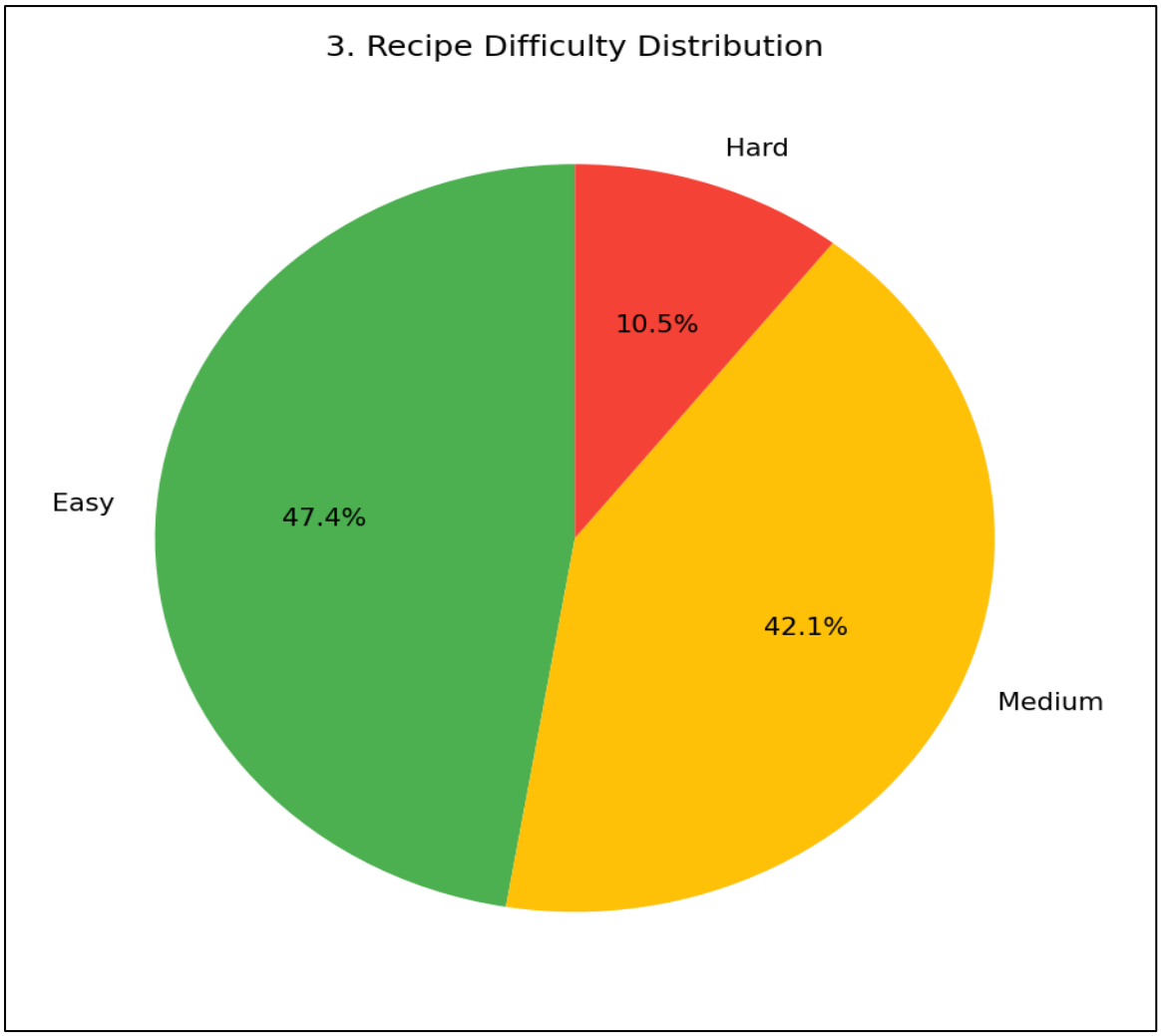
How: Computes the mean prep time and plots a histogram with a reference line for the average.

Key Observation (Qualitative): Most recipes fall into a **short to moderate prep window**, suitable for daily cooking. Idli Sambar sits in the mid-range with a 20-minute prep time, making it fairly accessible.

Why it matters:

- Helps position the platform: quick meals vs. elaborate weekend recipes.
- Opens up opportunities for filters like “Under 20 minutes” or “30-minute dinners”.

4.3 Difficulty Distribution (Easy / Medium / Hard)



What: Analyzes the share of recipes labeled as Easy, Medium, or Hard.

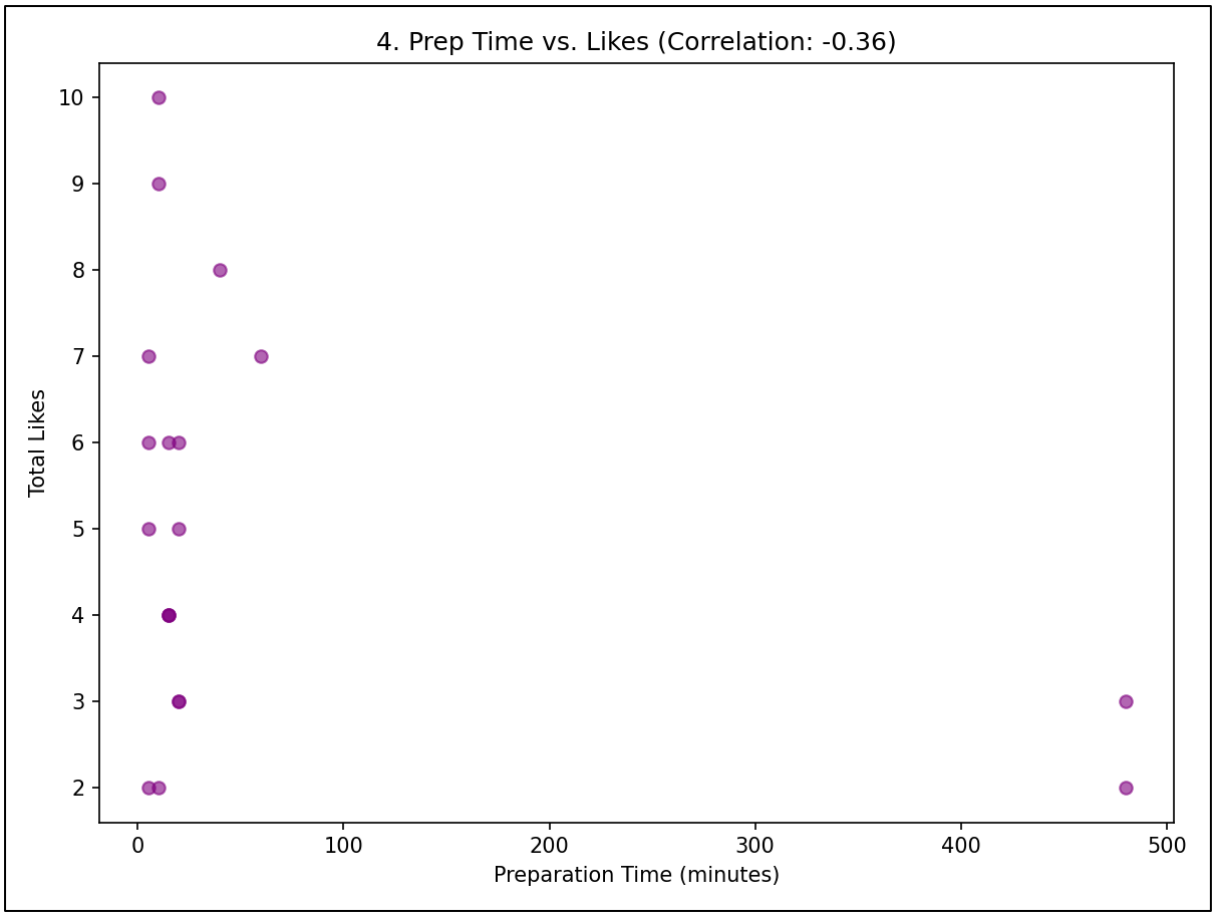
How: Aggregates difficulty from the recipes table and displays a pie chart.

Key Observation (Qualitative): The catalogue leans towards **Easy and Medium** difficulty, with a smaller subset of Hard recipes such as biryani-style or multi-step dishes. Idli Sambar is categorized as **Medium**, reinforcing its role as a flagship but approachable recipe.

Why it matters:

- Indicates that the current library is friendly to beginner and intermediate home cooks.
- Suggests potential gaps, e.g., adding more advanced dishes if targeting experienced cooks.

4.4 Correlation Between Prep Time and Likes



What: Explores whether recipes that take longer to prepare get more likes.

How:

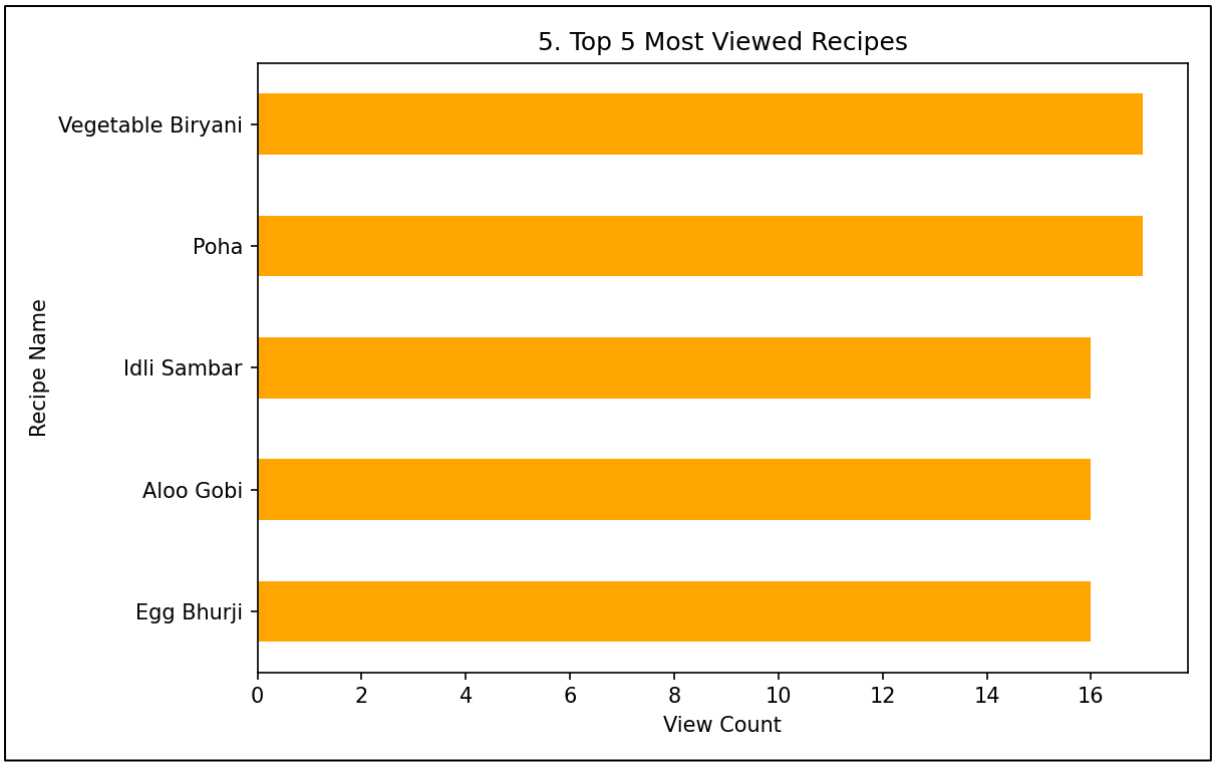
- Aggregates likes per recipe from clean_interactions.csv.
- Joins with prep_time_minutes from clean_recipe.csv.
- Computes Pearson correlation and visualizes via a scatter plot.

Key Observation (Qualitative): The correlation is modest, indicating that **longer prep time does not automatically guarantee more likes**. Simple comfort dishes often perform as well as or better than time-consuming recipes.

Why it matters:

- Encourages focusing on perceived value (taste, familiarity, convenience) rather than just complexity.
- Suggests that investing effort in “smart shortcuts” and one-pot recipes can be equally rewarding.

4.5 Most Frequently Viewed Recipes



What: Ranks recipes by total view count.

How: Filters interactions to type = "view", aggregates by recipe, and merges with recipe names. Outputs the top 5 and generates a horizontal bar chart.

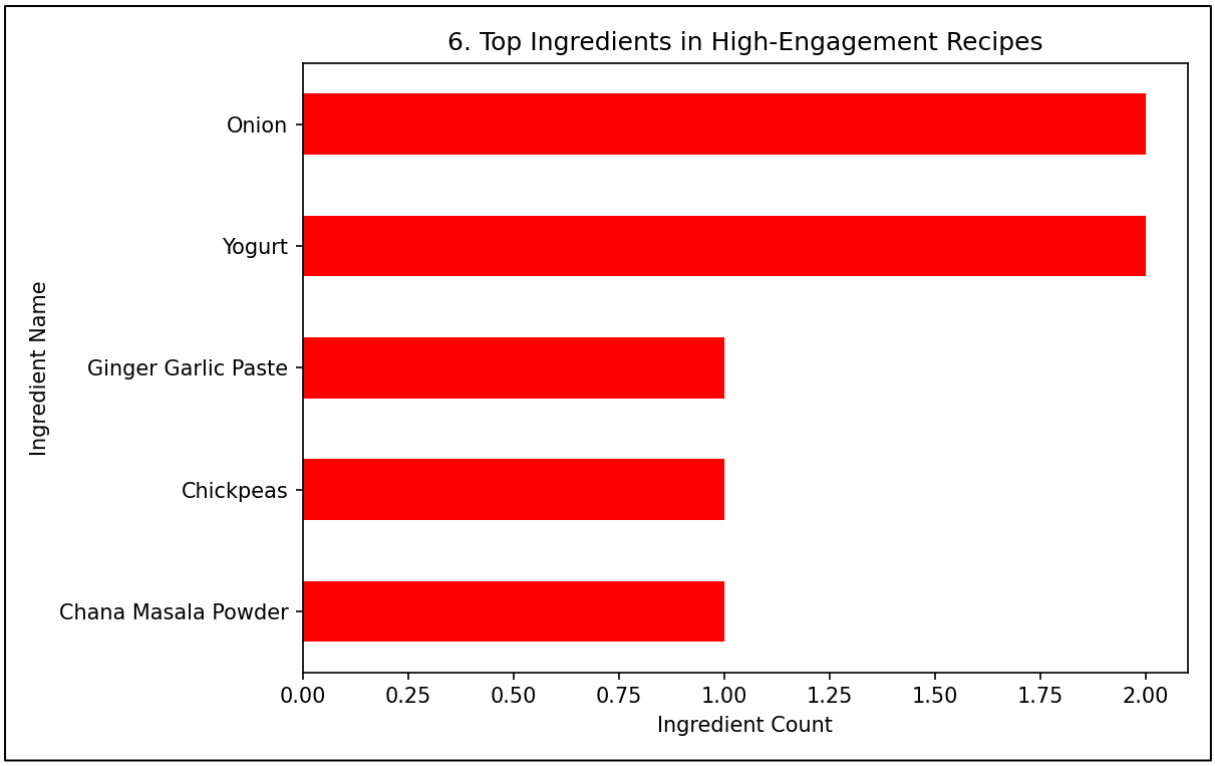
Key Observation (Qualitative): Visibility tends to cluster around:

- Popular everyday dishes (e.g., fried rice, paneer gravies).
- Breakfast staples like Idli Sambar and Poha.

Why it matters –

- Helps identify “hero recipes” suitable for highlighting on the homepage or in recommendation carousels.
- Can guide content strategy: similar recipes, variations, and “how-to” guides around high-traffic dishes.

4.6 Ingredients Associated with High Engagement



What: Finds which ingredients are most common in recipes with high engagement (likes, cook attempts, ratings).

How:

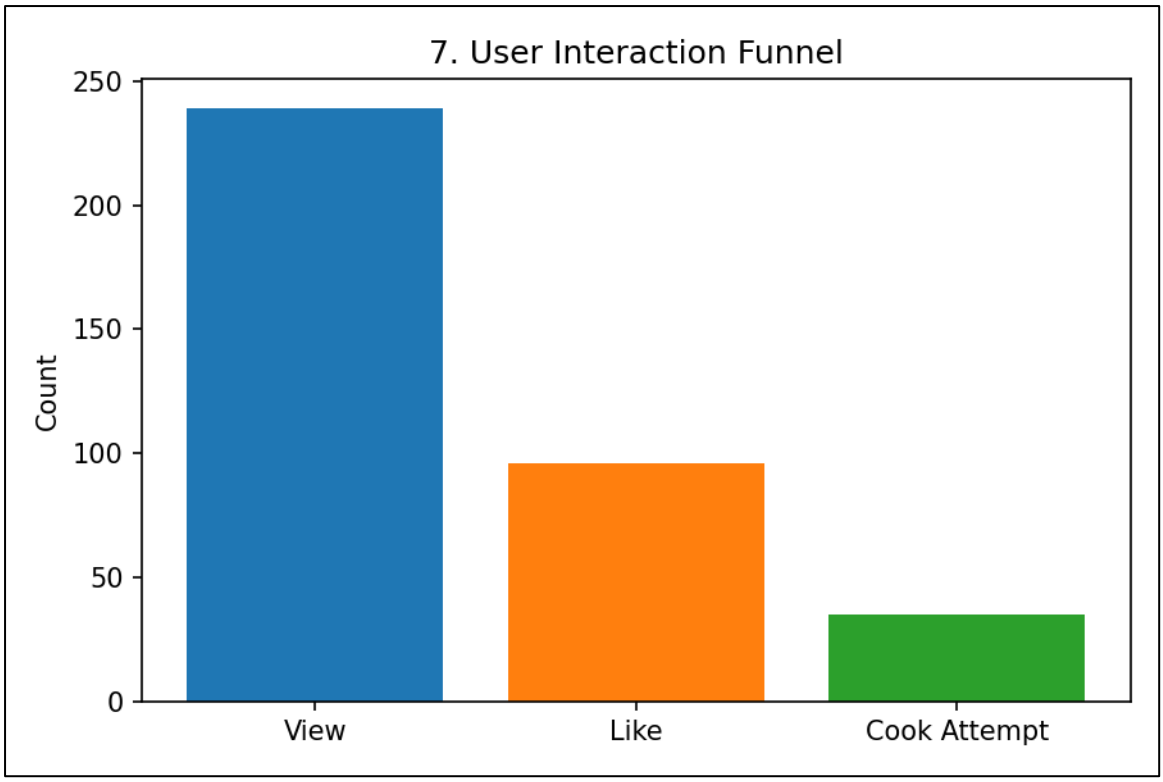
- Computes an engagement score per recipe using non-view interactions.
- Takes top-engagement recipes and counts their ingredients.
- Renders a bar chart of the top 5 “high-engagement ingredients”.

Key Observation (Qualitative): Ingredients such as paneer, rice, tomatoes, and certain spices appear frequently among high-engagement recipes, suggesting that **comfort and familiarity drive interaction**.

Why it matters:

- Guides future recipe curation (e.g., more paneer and rice-based recipes).
- Useful for seasonal campaigns, bundles, or cross-promotions with grocery partners.

4.7 User Interaction Funnel: View → Like → Cook Attempt



What: Summarizes the overall funnel from viewing a recipe to liking it and finally attempting to cook it.

How: Counts total events by type (view, like, cook_attempt) and visualizes them as a simple funnel bar chart.

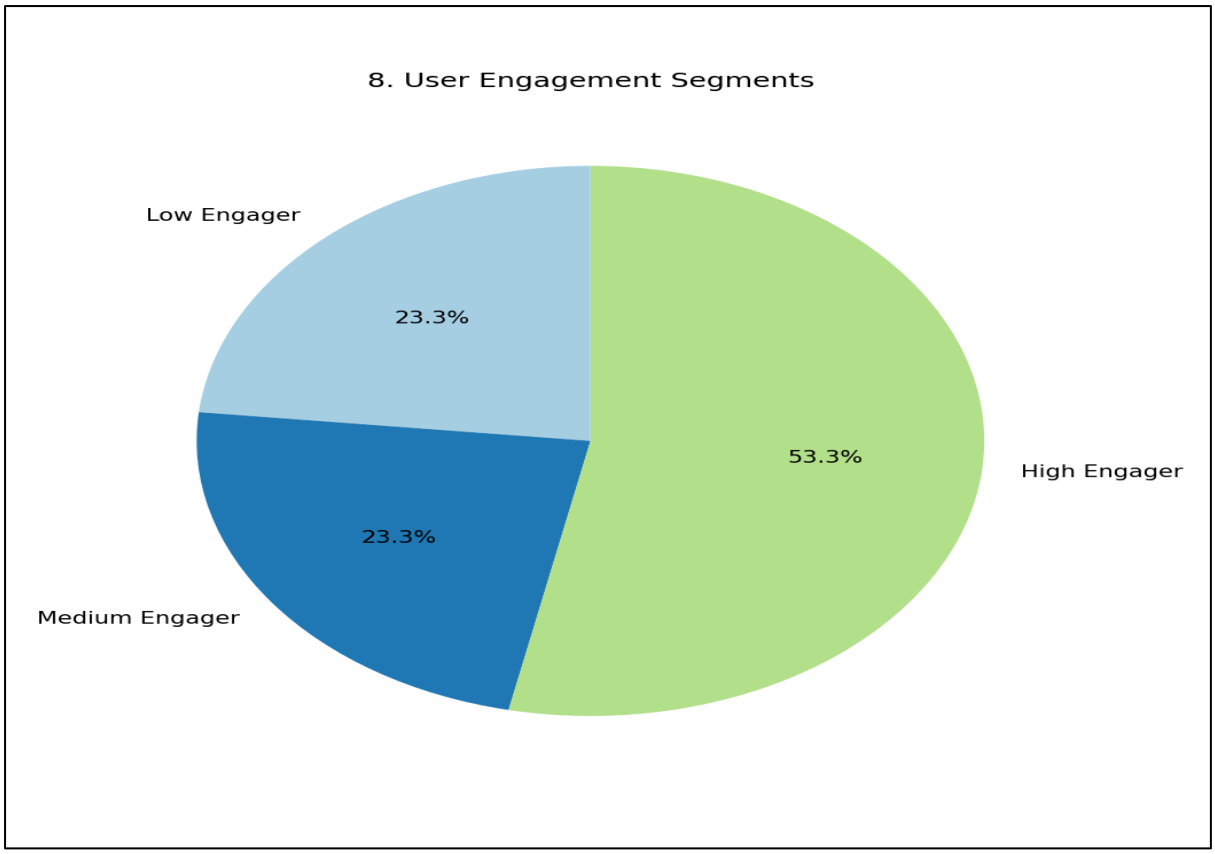
Key Observation (Qualitative):

- Views are highest, as expected.
- A subset of views convert to likes, and a further smaller subset convert to cook attempts.
- The drop-off between likes and cook attempts highlights the **"intent vs. action" gap**.

Why it matters:

- Provides a baseline conversion funnel.
- Insights can drive features like "shopping list generation", "smart reminders", or "cook later" prompts to nudge users from liking to actually cooking.

4.8 User Segments by Engagement Level



What: Clusters users into Low, Medium, and High engagers based on total interaction counts.

How:

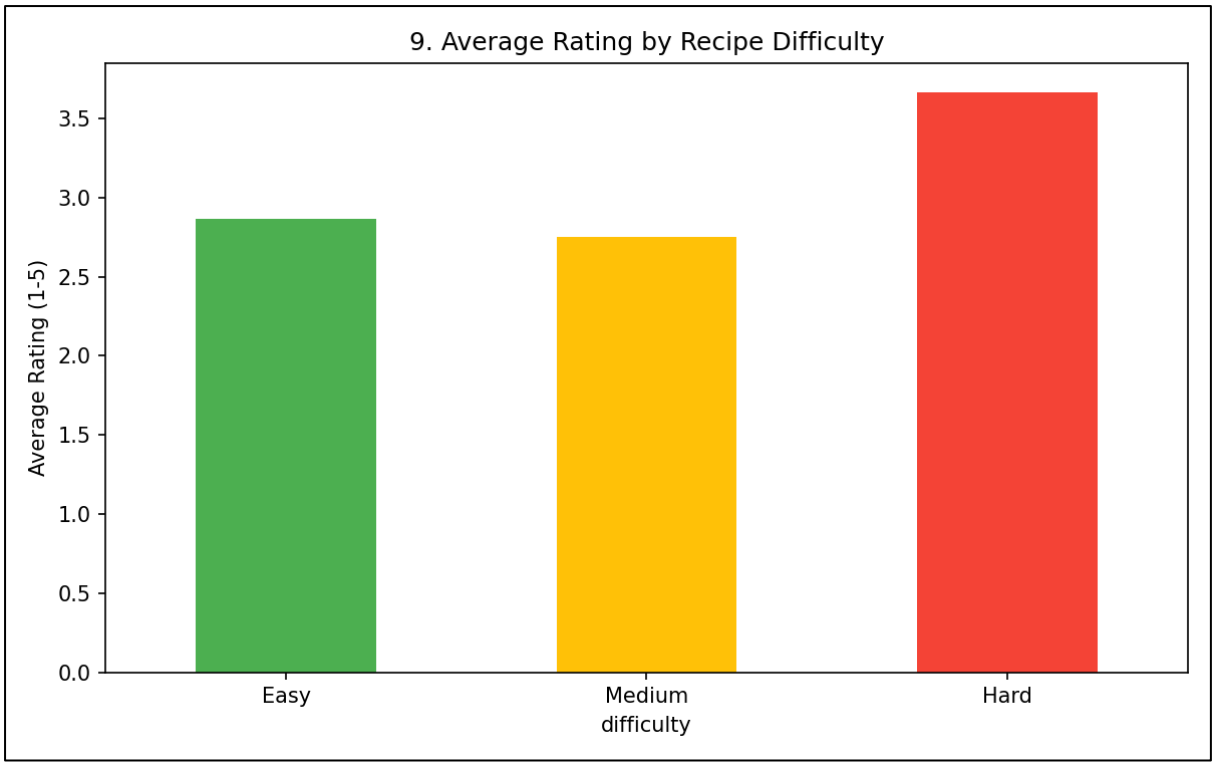
- Counts interactions per user.
- Uses quantiles (33rd and 66th percentiles) to segment into three tiers.
- Visualized with a pie chart showing segment proportions.

Key Observation (Qualitative): A small but significant cluster of **High Engagers** interact frequently (multiple likes/cook attempts/ratings), while a long tail of **Low Engagers** is present, typical of consumer platforms.

Why it matters:

- High engagers are strong candidates for beta features, feedback programs, and personalization.
- Low engagers may benefit from better onboarding, curated starter collections, or simplified navigation.

4.9 Average Rating by Difficulty



What: Compares average user ratings across Easy, Medium, and Hard recipes.

How:

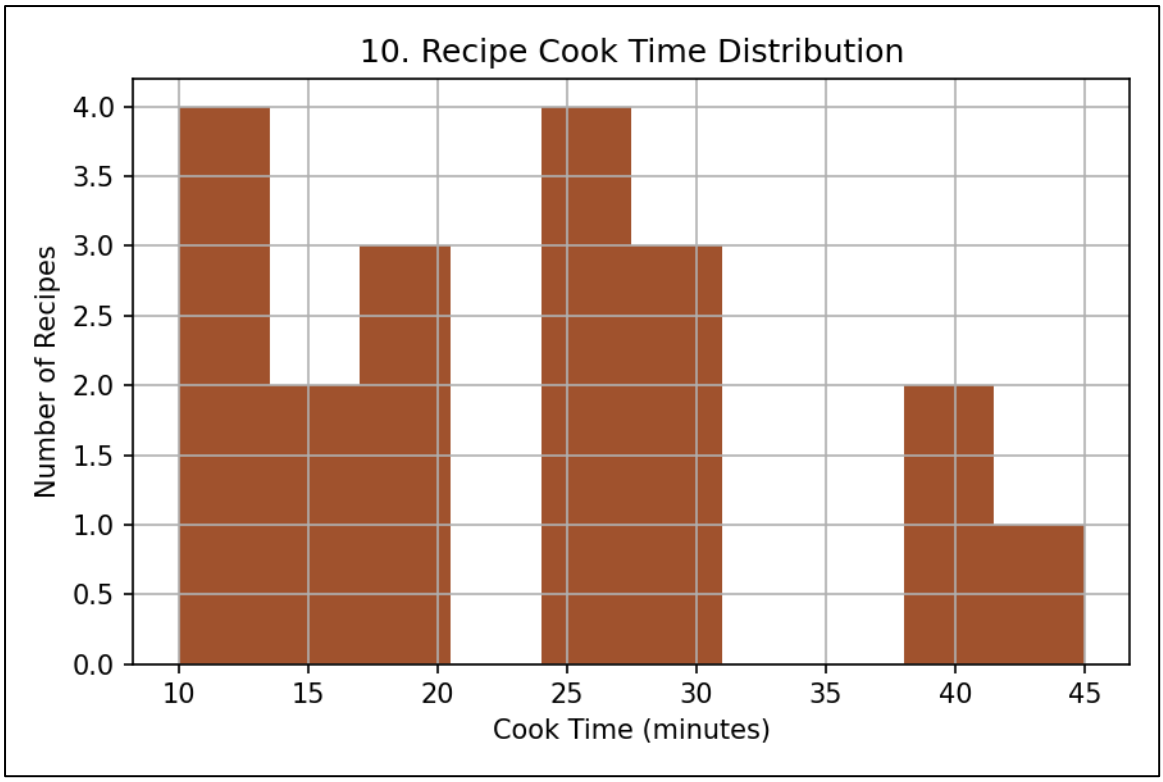
- Filters interactions to type = "rating".
- Joins with recipe difficulty.
- Computes average rating per difficulty and plots a bar chart.

Key Observation (Qualitative): Ratings across difficulty levels are relatively balanced, indicating that **well-executed recipes can perform strongly regardless of difficulty**, as long as expectations are clear.

Why it matters:

- Helps validate that Hard recipes are not punished unfairly by users.
- Supports adding more advanced recipes without fearing default lower ratings.

4.10 Cook Time Statistics



What: Analyzes cook_time_minutes distributions: min, max, mean, and spread.

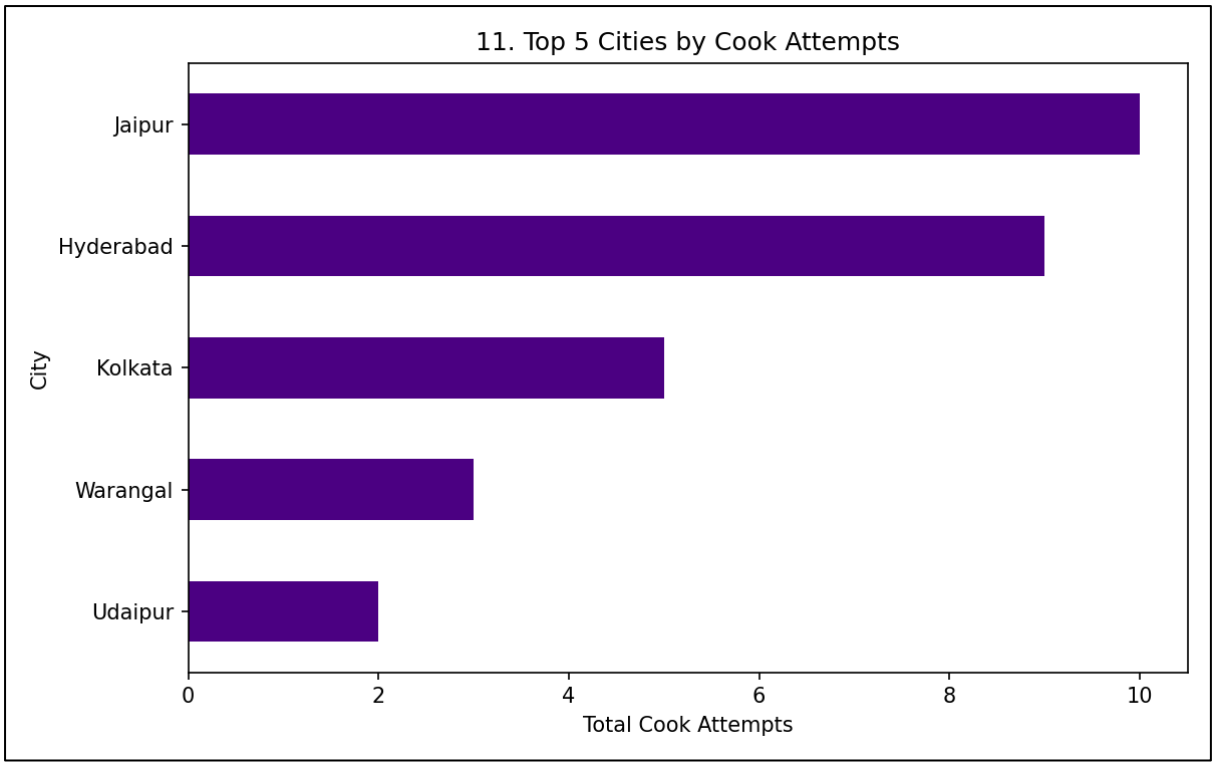
How: Uses summary statistics (describe()) and plots a histogram of cooking duration.

Key Observation (Qualitative): The catalogue offers a **good spread of quick and longer-cooking recipes**, from fast breakfasts like Upma or Egg Bhurji to slow-cook or layered dishes like Biryani and Rajma. Idli Sambar's 30-minute cook time positions it nicely as a **weekend or relaxed breakfast option**.

Why it matters:

- Enables time-based filtering (e.g., "Under 15 minutes").
- Helps product teams balance the mix of quick vs. elaborate recipes.

4.11 Top Cities by Cook Attempts



What: Identifies the top 5 cities with the highest number of cook attempts.

How:

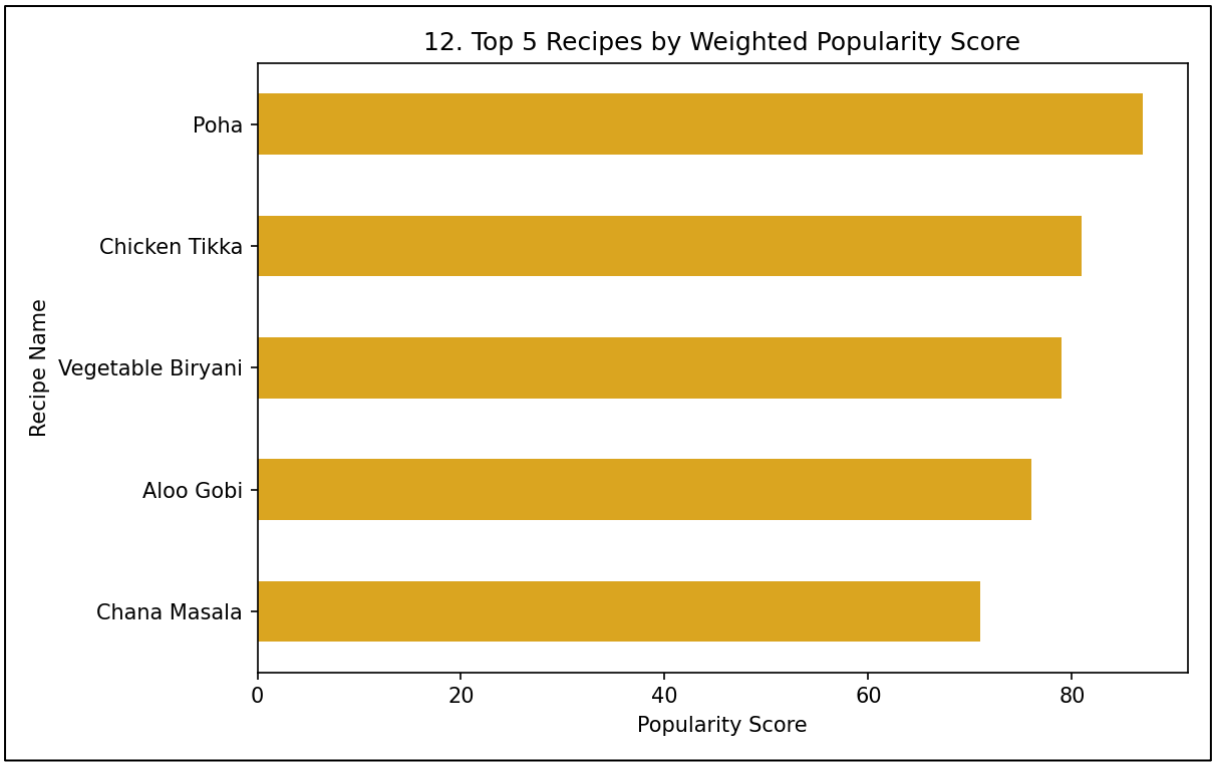
- Filters interactions to type = "cook_attempt".
- Joins with user metadata to attach city information.
- Aggregates attempts by city and plots a bar chart.

Key Observation (Qualitative): Cook attempts are concentrated in major metros such as Pune, Mumbai, Bengaluru, Chennai, etc., mirroring the seeded user base in the Firestore setup.

Why it matters:

- Supports regional marketing and localized content (e.g., Maharashtrian or South Indian focus where appropriate).
- Can be extended to build city-wise leaderboards or community features.

4.12 Weighted Popularity Score



What: Provides a composite popularity ranking for recipes using a weighted scoring scheme across all interaction types.

How:

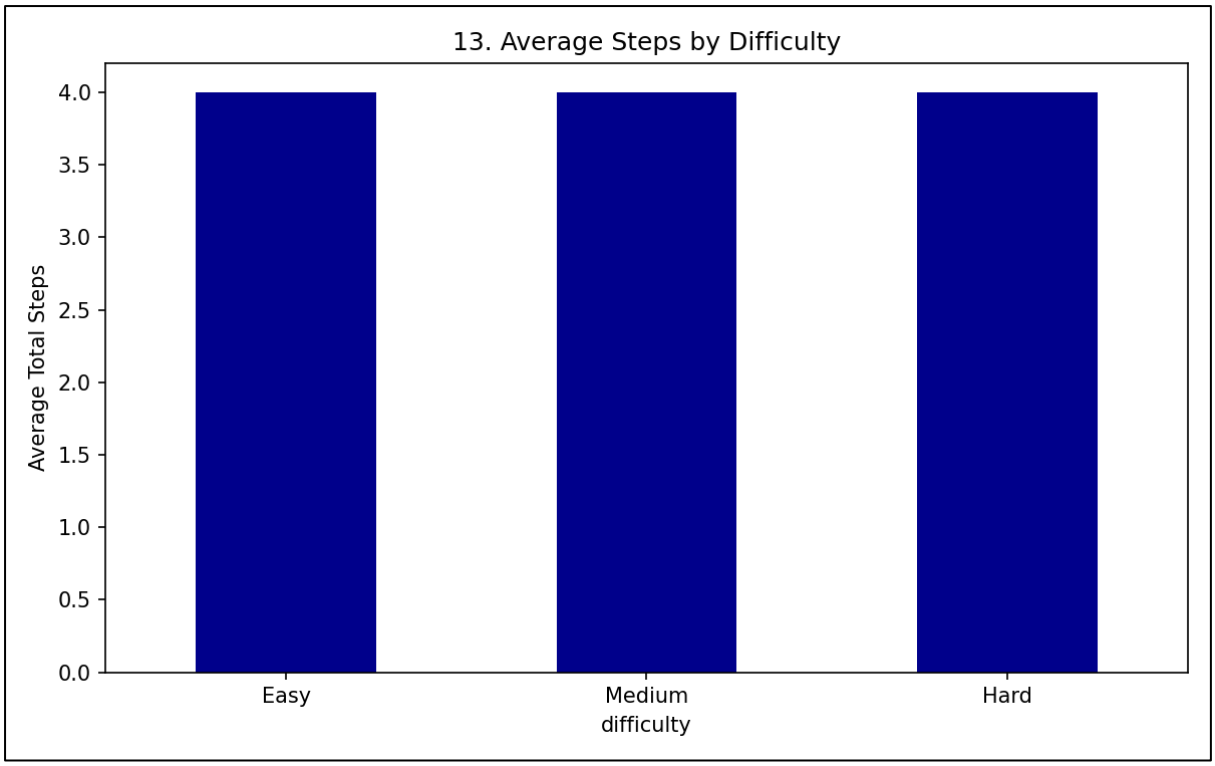
- Maps each interaction to a weight (view=1, like=5, cook_attempt=10, rating=2).
- Sums scores per recipe and ranks them.
- Produces a bar chart of the top 5 recipes by popularity score.

Key Observation (Qualitative): Some recipes with moderate views but **high cook attempts and likes** can outrank pure "view magnets", giving a more realistic sense of true engagement.

Why it matters:

- Helps pick genuinely successful recipes for featuring.
- Avoids over-indexing on vanity metrics like views alone.

4.13 Average Steps by Difficulty



What: Examines whether harder recipes actually have more steps.

How:

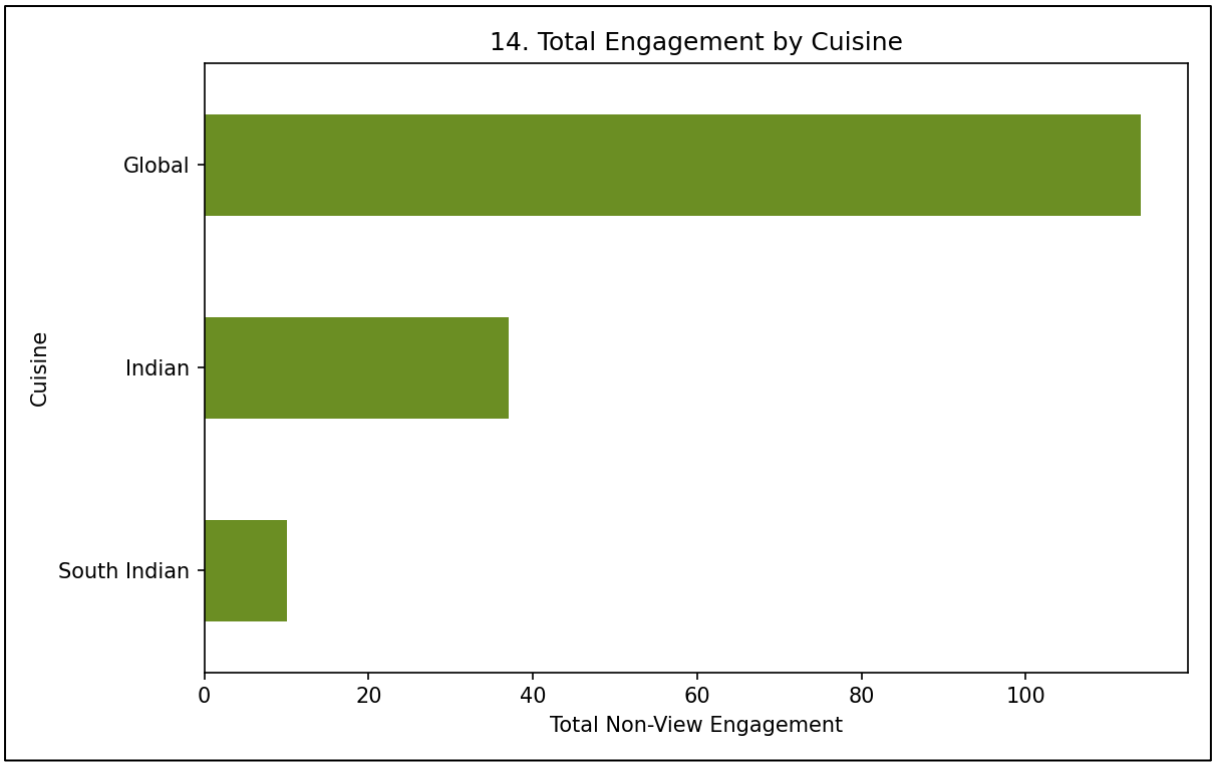
- Uses clean_steps.csv to compute total steps per recipe.
- Joins with difficulty from recipes and calculates average steps per difficulty group.
- Visualized as a bar chart.

Key Observation (Qualitative): As expected, **Hard recipes tend to involve more steps**, while Easy recipes are shorter and more linear. Medium recipes, including Idli Sambar, strike a balance between structure and simplicity.

Why it matters:

- Validates that difficulty labeling is consistent with actual recipe structure.
- Useful when designing UX features like step timers or guided cooking modes.

4.14 Engagement by Cuisine



What: Evaluates which cuisines generate the most non-view engagement (likes, cook attempts, ratings).

How:

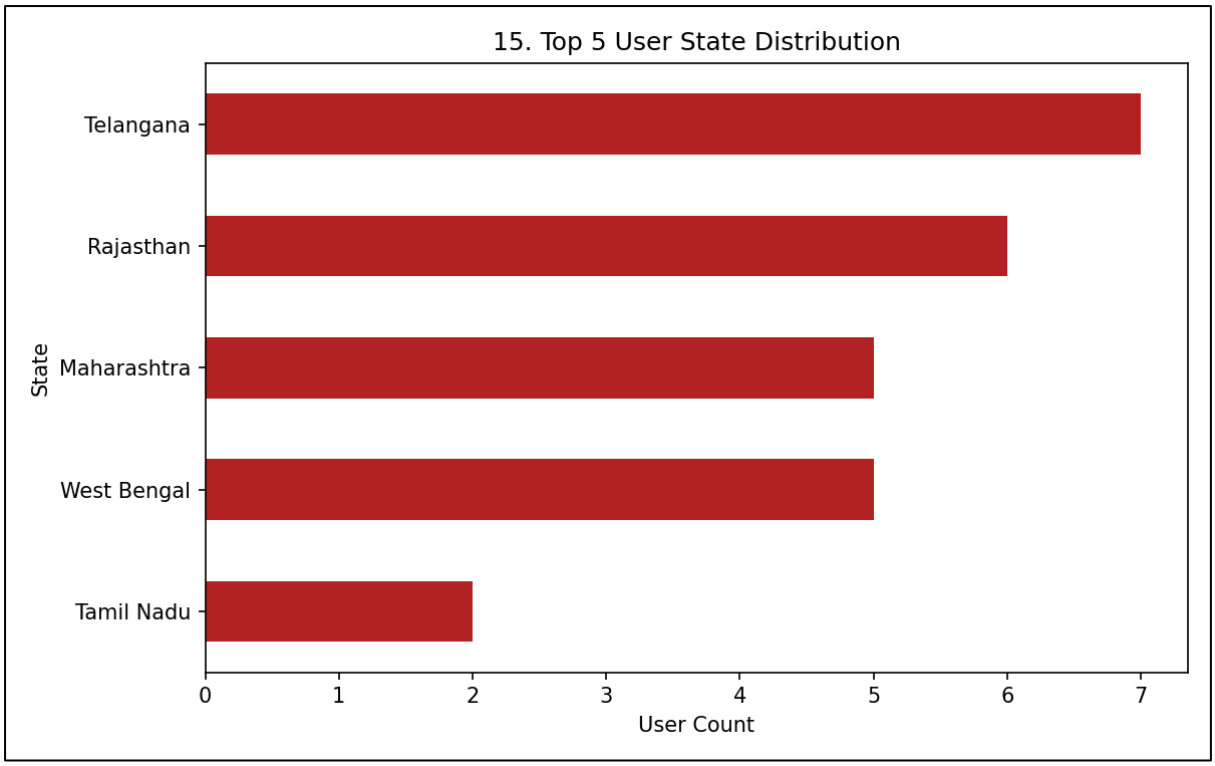
- Filters interactions to non-view types.
- Aggregates engagement by recipe, then explodes the cuisines array and sums engagement by cuisine.
- Renders a bar chart of the top cuisines.

Key Observation (Qualitative): Indian sub-categories (North Indian, South Indian, Comfort/Home-style) show strong engagement, with "global" dishes (e.g., pasta, fried rice) also performing well as variety options.

Why it matters:

- Provides direction on which cuisine lines to deepen.
- Informs future expansion into under-served cuisines that show promising early engagement.

4.15 User State Distribution



What: Identifies the top 5 states by user count.

How: Aggregates state from clean_users.csv and plots a bar chart.

Key Observation (Qualitative): States such as **Maharashtra, Karnataka, Tamil Nadu, Delhi, Telangana, and Gujarat** are well represented, consistent with the seeded location model in firestore_setup.py.

Why it matters:

- Helps plan regional rollouts and language localization.
- Can be tied to state-wise festivals and seasonal recipe recommendations.

5. Outputs Produced

The analytics stage generates the following artifacts:

- **JSON Summary**
  - Analytics_Output/analytics_report.json
  - Contains machine-readable structures for all 15 insights (top lists, distributions, funnel counts, correlation values, etc.).
- **Tabular Outputs**
  - Interaction summaries (per recipe, per type).
  - User engagement summaries (interactions per user).
  - Popularity scores merged with recipe metadata.
- **Charts**
  - 15+ PNG charts under Analytics_Output/Charts/, including:
    - Ingredient frequencies
    - Preparation and cook time distributions
    - Difficulty, segments, and state pies
    - Top recipes by views and popularity
    - Engagement by cuisine and city
    - Prep time vs. likes scatter plot

These assets together form both a **human-readable insight layer** and a **machine-consumable analytics layer** that can be plugged into dashboards or reporting tools.

6. Limitations & Future Enhancements

Current Limitations

- The dataset is synthetic, realistic, and optimized for showcasing pipeline behavior rather than reflecting production-scale traffic.
- Interactions are generated in a controlled way, so patterns (e.g., city/state distribution) mirror the seeded configuration rather than organic user growth.
- Time-series aspects such as seasonality or trend analysis are not deeply explored in this version.

Potential Enhancements

- Introduce time-based analyses (daily/weekly active cooks, trending recipes).
- Integrate cost or nutrition dimensions for value-based recommendations.
- Expose the analytics outputs to a BI layer (e.g., BigQuery + dashboard) for interactive slicing and dicing.
- Scale the pipeline to handle larger datasets and more complex event types (shares, comments, collections).

7. Conclusion

The analytics layer of this Firebase-based recipe platform demonstrates a **complete, end-to-end journey** from raw Firestore events to clean, validated data and finally to rich behavioral insights. By combining a structured data model, robust validation rules, and a focused analytics engine, the solution provides a clear view into **which recipes work, how users interact, and where future product opportunities lie** with Idli Sambar anchored as the primary reference recipe at the heart of the dataset.