Restaurant Recommendations & Clustering with RFM Analysis- Model Deployment

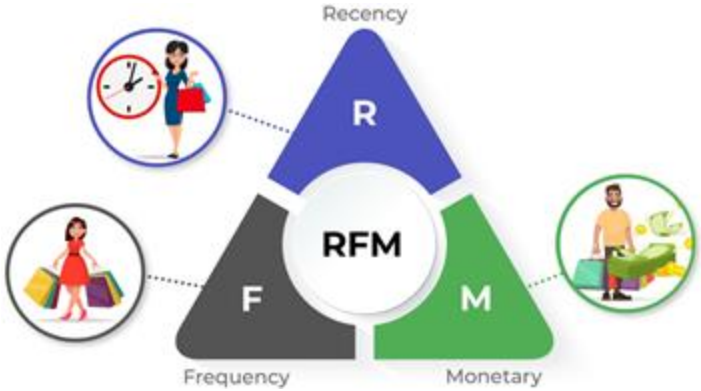
Team Members:
Glenys Lion
Fuqian Zou
Iris Lee
Kavya Bhat
Liana Bergman-Turnbull

Northwestern | McCORMICK SCHOOL OF ENGINEERING

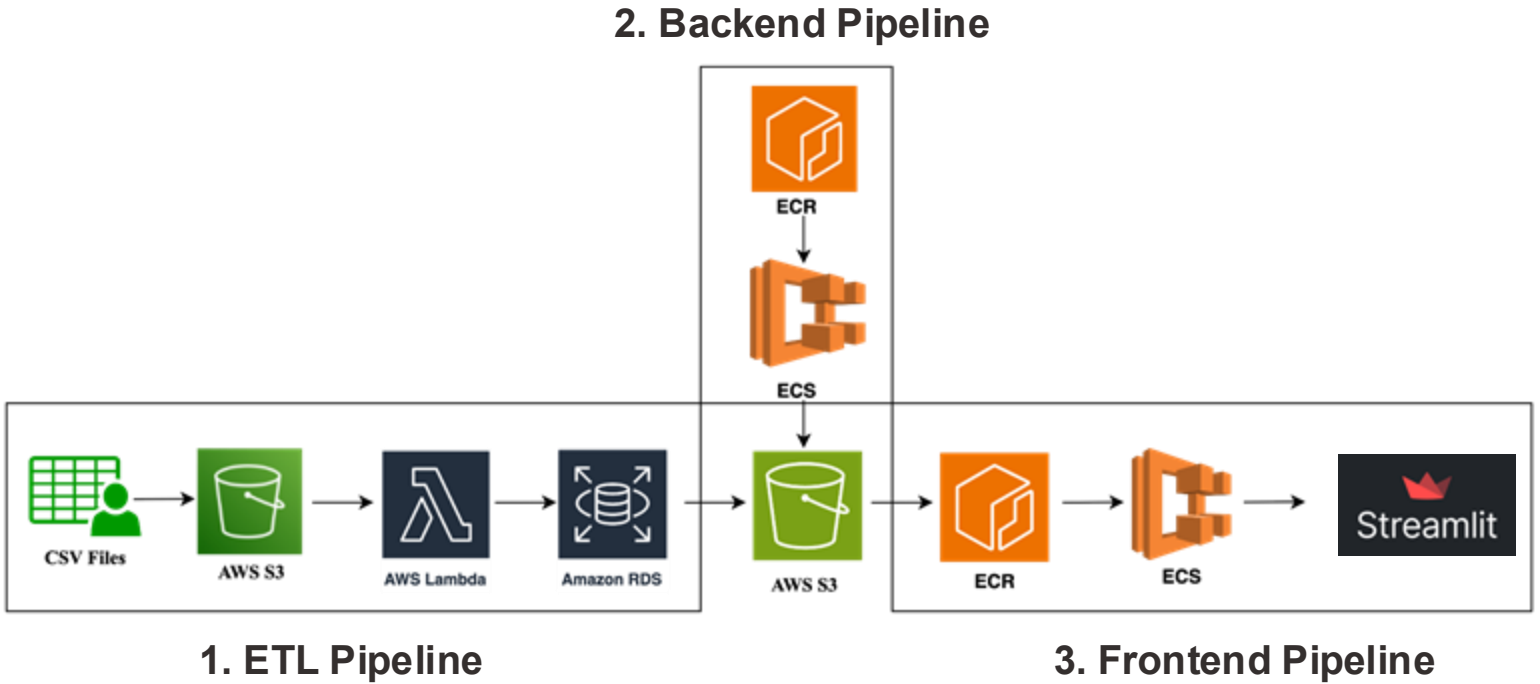
Table of Contents

Problem Statement

- We implemented a clustering algorithm, and built a recommendation system to help pair users with vendors benefiting both users and vendors. Our model was integrated into a front-end application and deployed using AWS, providing a seamless and scalable solution for personalized restaurant discovery.



Cloud Pipeline Workflow



Budget Estimation

AWS Service Name	Average Monthly Cost($)
AWS Lambda	0.00
RDS for PostgreSQL	73.11
AWS Train Service (Fargate)	1.50
AWS Fronted Service (Fargate)	36.04
Elastic Container Registry	0.38
Simple Storage Service (S3)	0.43
CloudWatch	13.16

Average Yearly Cost:
1,495.44 USD

Estimated URL:
https://calculator.aws/#/estimate?id=4d3f2a9bf862f0bec8c633ed4489a78b0ddee63a

Cloud-Based Data Ingestion & ETL Pipeline

- Raw data files (orders, products, customers, payments) collected from kaggle

- Used as a data lake to store all four raw CSV files.
- Files were uploaded manually.

- ETL via Lambda function was triggered to clean, transform and merge the CSVs into one unified file

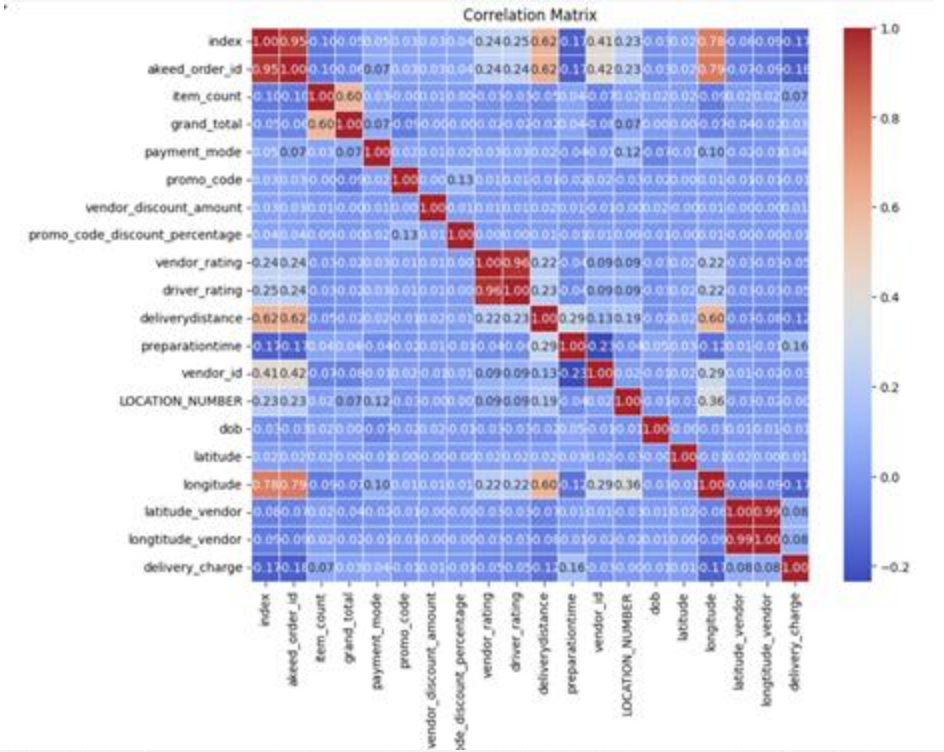
- A PostgreSQL database used to store the final clean dataset for further querying
- pgAdmin was used to verify and explore the cleaned data stored in RDS

- Once stored in RDS, the clean data is available for further analysis, visualization and downstream ML Pipelines.

EDA

- A moderately strong positive correlation between item count and total cost
- A moderately strong positive correlation between delivery distance and longitude
- Most features are not highly correlated

AWS: EDA plots were uploaded to S3 bucket



Customer Segmentation - RFM Analysis

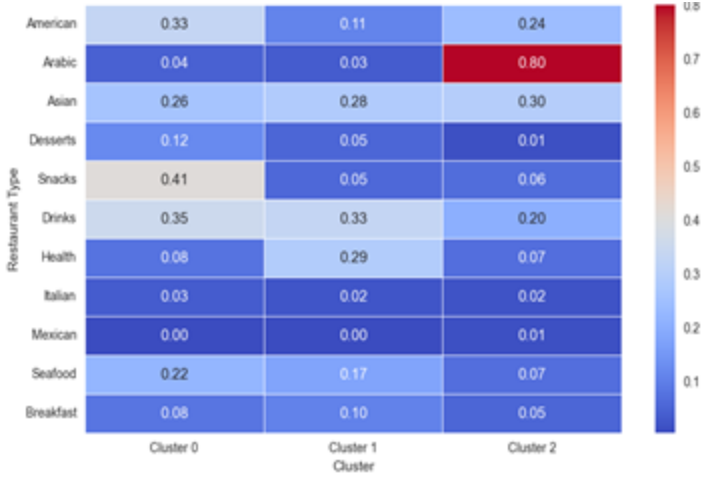
- **Objective:** Segment users based on purchasing power
- **Data Preprocessing:** create features (recency, frequency, monetary)
- **Train Model:** Scale Monetary by logging, run K-means (k=4), map clusters appropriate segment names
- **Customer Lifetime Value (30 days):** For each segment, calculate the average orders per month x average order value (monetary / frequency) per month
- **Output**: Dataframe with Customer ID,segment,CLV_30

Cluster	Recency (days)	Frequency (# of orders)	Monetary (US dollars)	CLV / Month (US dollars)	Proportion (% of users)	Description
1	17	30	$493.57	$70.76	6%	Super User
2	37	7	$114.70	$31.21	34%	Regular User
3	45	2	$17.85	$19.55	38%	Churn User
4	189	2	$26.71	$12.56	22%	Lost User

Customer Segmentation - Food Analysis

- **Objective:** Segment users based on food preferences
- **Data Preprocessing:** explode vendor tags into dummy columns, aggregate based on broader food categories, sum tag counts for each customer per category
- **Train Model:** applies TF-IDF scaling to normalize the frequency matrix, runs K-Means (k=3), map clusters to appropriate food segment (western, asian, arabic)
- **Output**: Dataframe with Customer ID, Food Segment

Upload to AWS: Join two dataframes on customer ID, upload to S3 in CSV



K-Means Heatmaps
Cluster 0: Western, Cluster 1: Asian, Cluster 2: Arabic

Recommendation System Models

- **Data Preprocessing:**
  - Group customers to calculate the total number of orders for each customer at each restaurant
  - Ratings are based on the order frequency

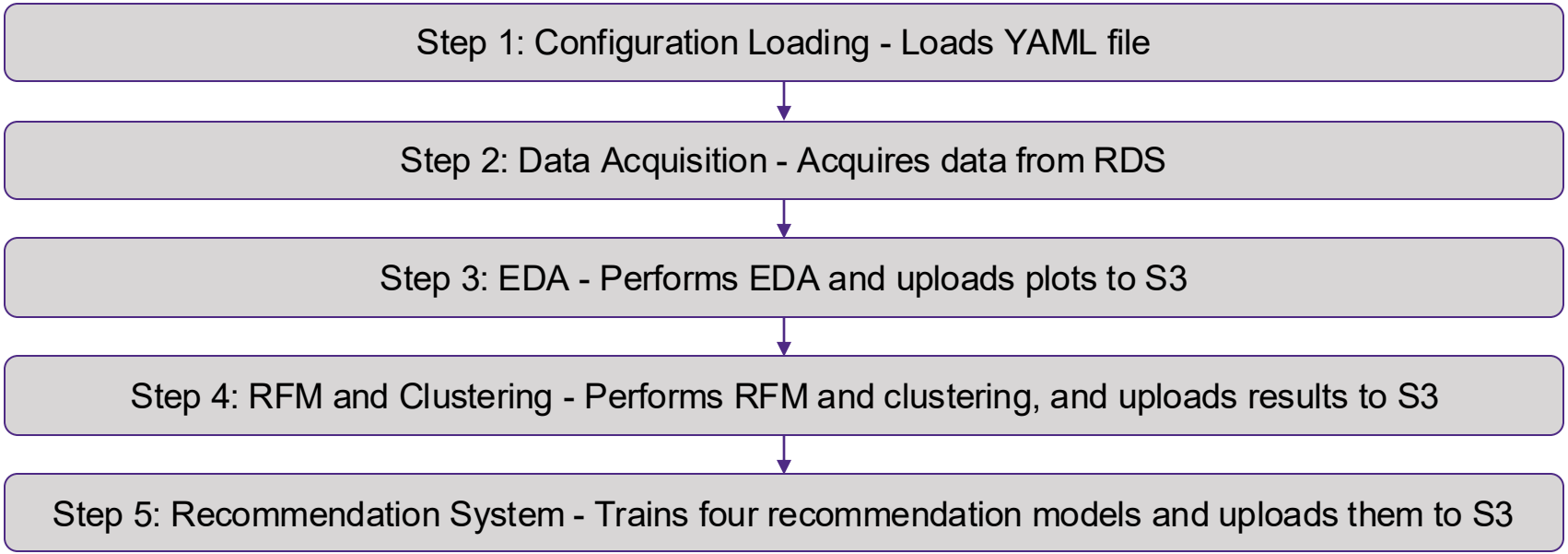
- **Models Implemented:**

Model Name	Parameters	RMSE
Item-Based CF	K: 30, Min K: 10, Similarity Options: cosine	2.8121
SVD	n_factors: 100, n_epochs: 20, lr_all: 0.005, reg_all: 0.1	2.6002
NMF	n_factors: 50, n_epochs: 15, lr_all: 0.1, reg_all: 0.1	2.6802
SVD++	n_factors: 20, n_epochs: 10, lr_all: 0.002, reg_all: 0.05	2.6054

- **Top-K Recommendation:**
  - Filters out unrated restaurants for the customer
  - Predicts ratings for unrated restaurants
  - Ranks and selects the top K restaurants with the highest predicted ratings

Backend Pipeline

- **Pipeline Overview:**

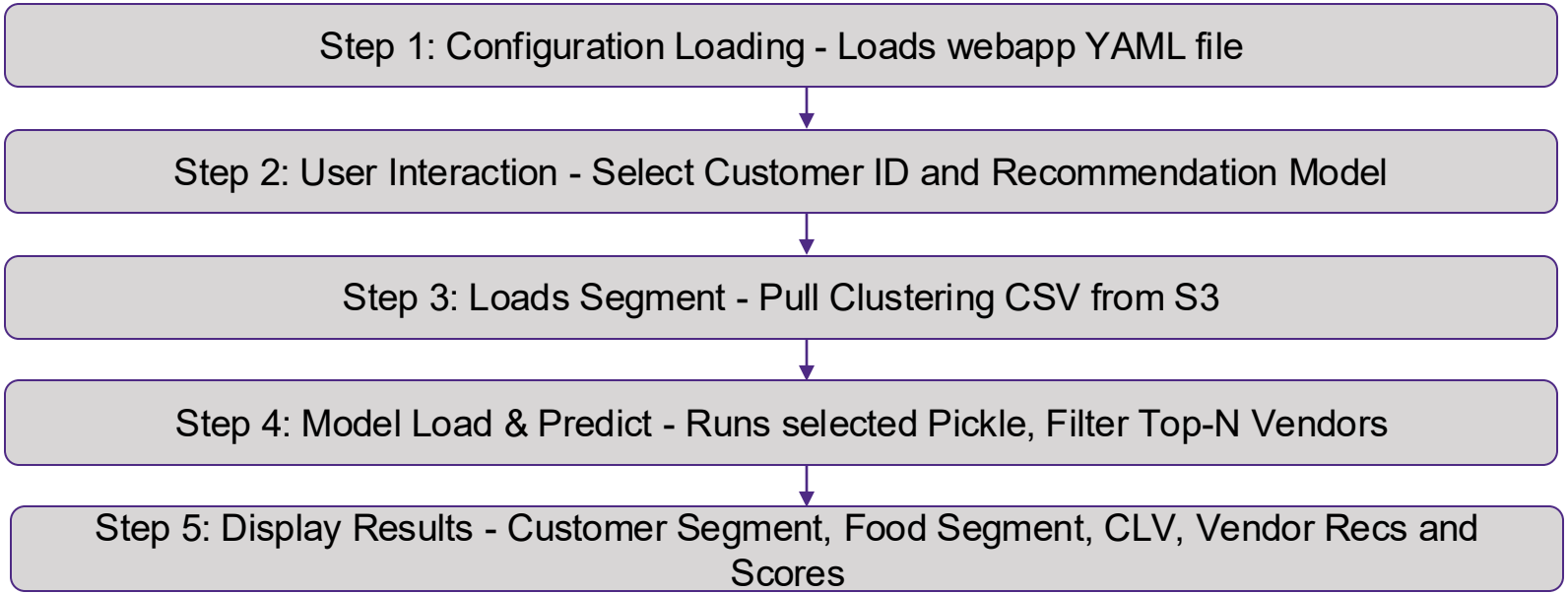


Backend Pipeline

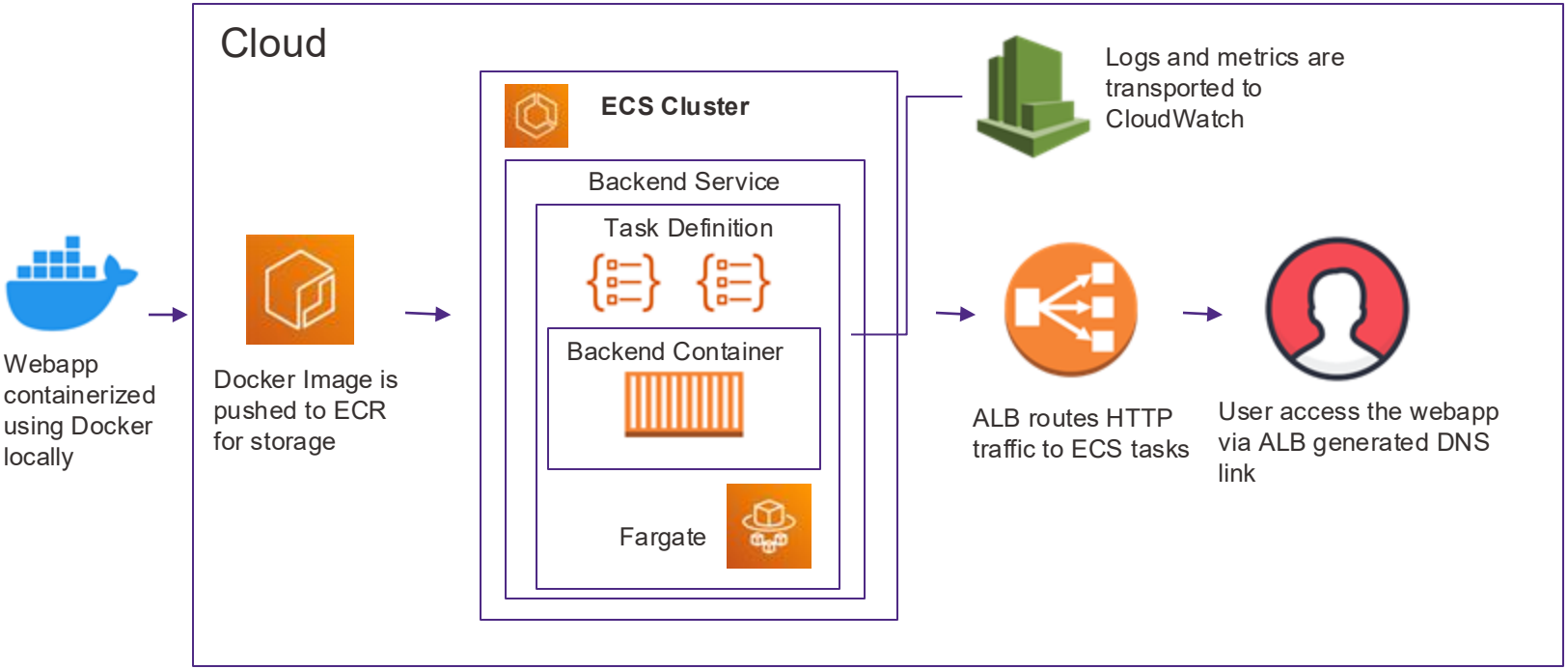
- **Logging:** All modules record detailed logs for easier monitoring and debugging.
- **Exception Handling:** Robust mechanisms in place to catch and handle errors gracefully.
- **Unit Testing:** 19 unit tests implemented, covering all key modules for reliability.
- **AWS Credentials Handling:** AWS credentials are securely stored in a .env file, which is included in .gitignore to prevent accidental leaks.
- **Configuration:** Centralized YAML file manages model parameters, feature selection, and AWS settings.
- **Dockerization:** The entire backend pipeline is containerized for consistent deployment and portability.

Frontend Pipeline

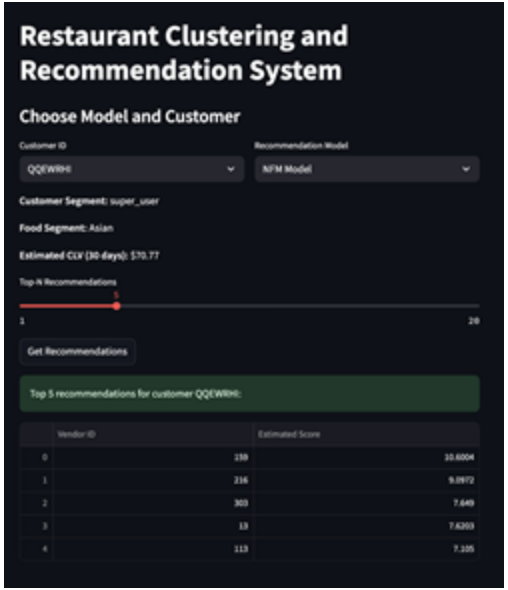
- **Pipeline Overview:**



Frontend System Workflow



Frontend UI



AWS Deployed Webapp: http://cluster-recommendation-frontend-1670942115.us-east-1.elb.amazonaws.com/

* **Notes:** We turned off the AWS Services after doing the video recording.

Northwestern | McCORMICK SCHOOL OF ENGINEERING

Thank You

Northwestern | McCORMICK SCHOOL OF ENGINEERING