COMP4801 FYP Final Presentation
E-commerce Web App for Toys

Group: fyp24112 Choi Tsz Long 3035705154

TABLE OF CONTENTS

01

Introduction

Project Background (Recap)

- **Great Potential for HK's Toy Market**:
  - HK being one of the largest toy importers in the world – 7th largest in 2019, >1.5M toy import trade value
  - High Popularity and Business Volume of ACGHK – shows high influential power of toys to HK people and their high purchasing power for toys

Reporter	TradeFlow	ProductCode	Product Description	Year	Partner	Trade Value 1000USD	Quantity	Quantity Unit
United States	Import	950390	Toys nes	2019	World	15,387,285.54	1,193,500,000	Kg
European Union	Import	950390	Toys nes	2019	World	8,591,127.78	893,532,000	Kg
Germany	Import	950390	Toys nes	2019	World	3,399,220.63	243,803,000	Kg
United Kingdom	Import	950390	Toys nes	2019	World	2,728,911.39	267,594,000	Kg
Japan	Import	950390	Toys nes	2019	World	2,299,262.84	138,771,000	Kg
France	Import	950390	Toys nes	2019	World	2,254,185.19	189,233,000	Kg
Hong Kong, China	Import	950390	Toys nes	2019	World	1,529,598.23	170,335,000	Kg
Canada	Import	950390	Toys nes	2019	World	1,407,684.34	107,270,000	Kg
Russian Federation	Import	950390	Toys nes	2019	World	1,318,132.75	153,210,000	Kg
Netherlands	Import	950390	Toys nes	2019	World	1,308,962.62	127,793,000	Kg
Italy	Import	950390	Toys nes	2019	World	1,183,013.45	91,256,700	Kg
Czech Republic	Import	950390	Toys nes	2019	World	1,127,745.58	141,757,000	Kg

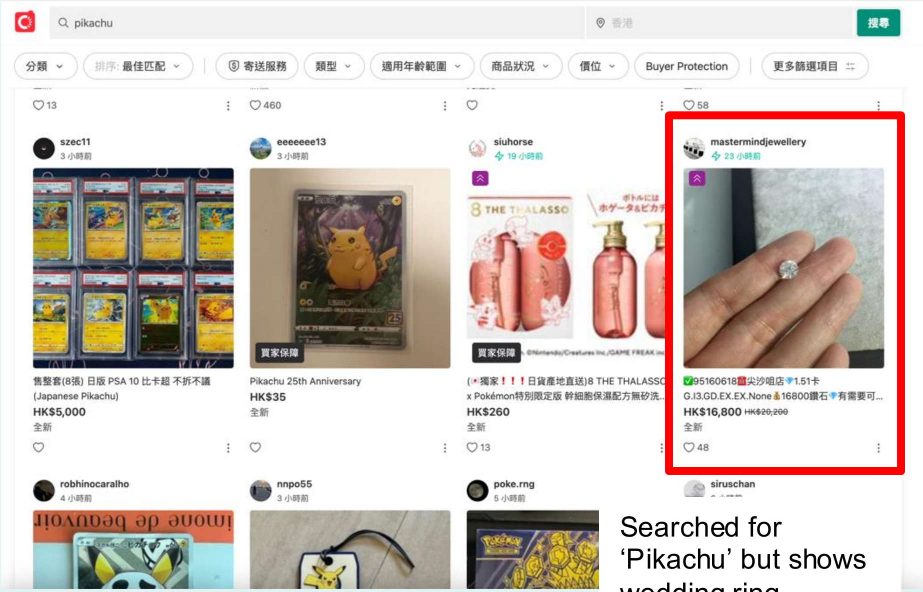
Project Background (Recap)

- **However, the market's potential is limited**:
    - Inadequate channels to buy/sell toys online in Hong Kong
    - Most local toy stores do not have their own online store, only sell toys at Brick-and-Mortar stores/rely on other E-commerce Platforms
    - Insufficient market information leading to Information Asymmetry
    - Market ecology damaged by speculative behaviour -> consumers lose confidence

Project Background (Recap)

- **Another Problem...**
    - Existing Mainstream E-commerce Platforms are not specialized in toys or not localized for HK -> User experience is not optimal, E.g. Carousell, Taobao, Amazon

Project Background (Recap)

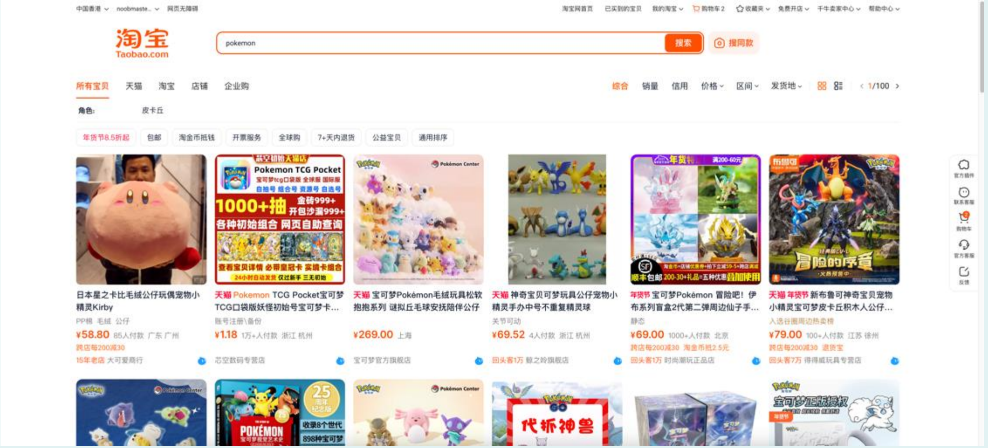


Searched for 'Pikachu' but shows wedding ring

Example 1: Carousell

- Allows user to sell all kinds of products and services, not limited to toys
- Too many product categories -> <u>lowers the accuracy of product recommendation engine</u>, irrelevant recommendation can often show up

Project Background (Recap)



Example 2: Taobao

- Services are not localized for HK (No free shipping for many sellers)
- Complicated website structure, e.g. Taobao vs World Taobao vs Tmall?

Project Objective

Include all essential functionalities for Both Buying & Selling Toys (Support Both B2C and C2C)

User friendly UI/UX Optimised for Toys Trading

Objective: To Create A One-stop E-commerce Web App for Toys

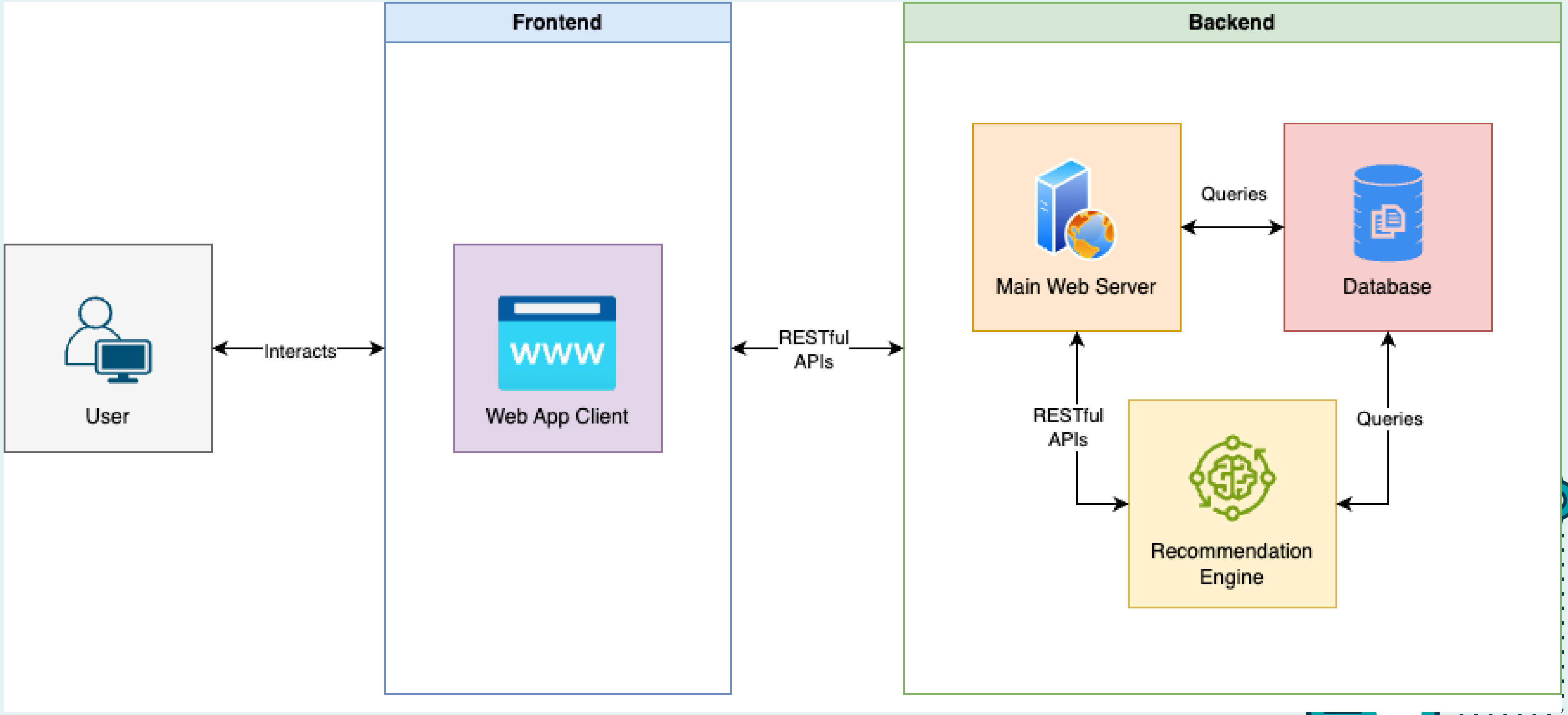
Easy access to market information -> avoid speculation

Personalized Product Recommendation with High Accuracy

02

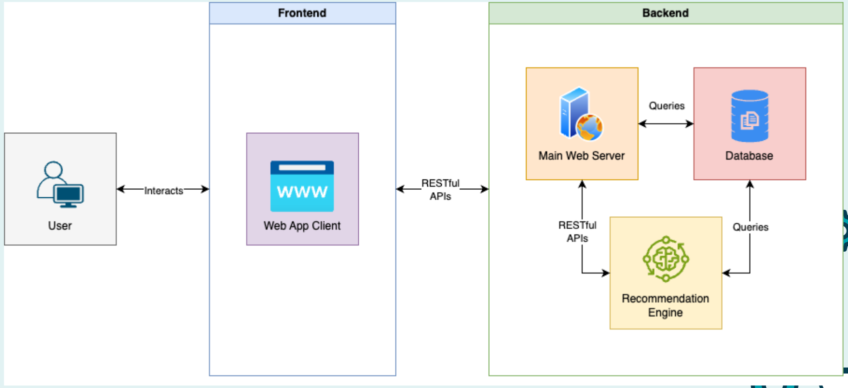
Project Methodologies

System Architecture



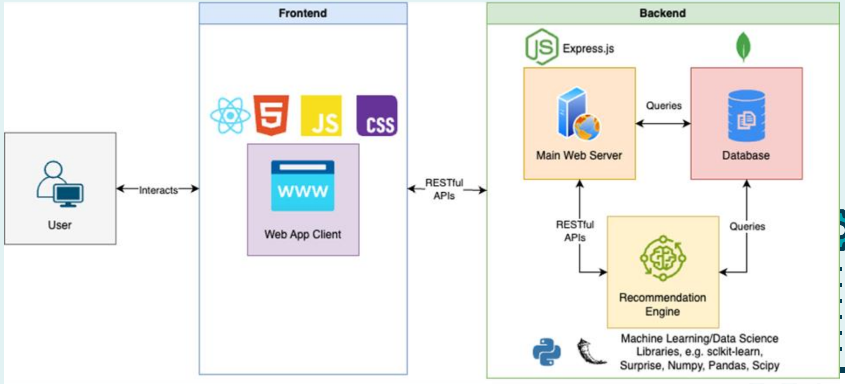
System Architecture - Breakdown

- **Frontend**
  - **Web App Client**: Directly interacted by users; can be broken down to pages -> components -> services
- **Backend**
  - **Main Web Server**: manipulates the data from the Frontend and other Backend components to operate the major features of the web app, e.g. product management, User authentication, user interaction tracking
  - **Recommendation Engine + server**: generates personalized recommended products for each user and display them on the corresponding product sections on the web app client
  - **Database**: stores all data for the app, e.g. users, products, product reviews, user events

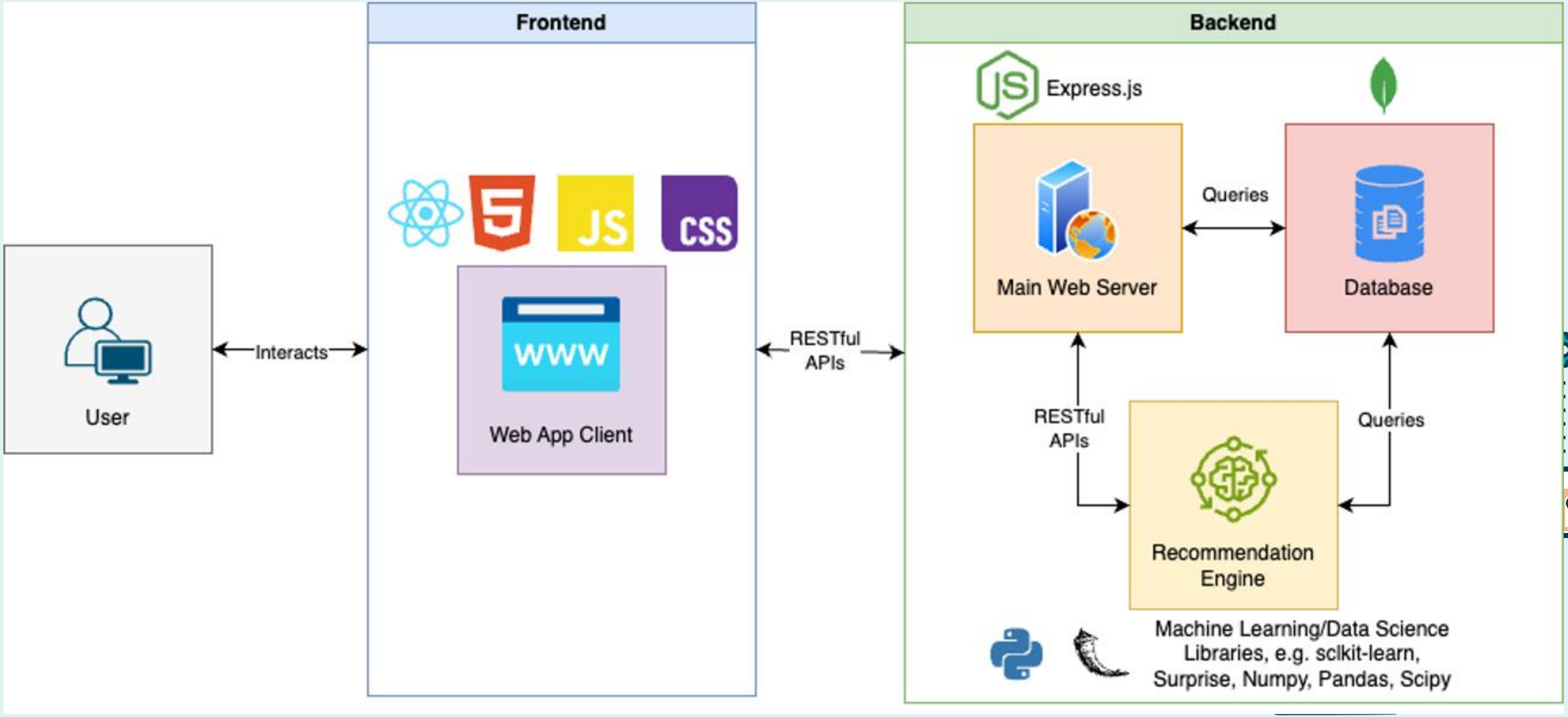


Software Specs

- **Frontend**
  - **Web App Client**: React JS (HTML, CSS, JavaScript)
- **Backend**
  - **Main Web Server**: Node JS, Express JS
  - **Recommendation Engine + server**:
    - Server: Python Flask
    - Data processing + Model training: ML and DS libraries, e.g. sclkit-learn, Surprise, Numpy, Pandas, Scipy
  - **Database**: MongoDB
- RESTful API for **data exchange** between Front and Backend/Backend components

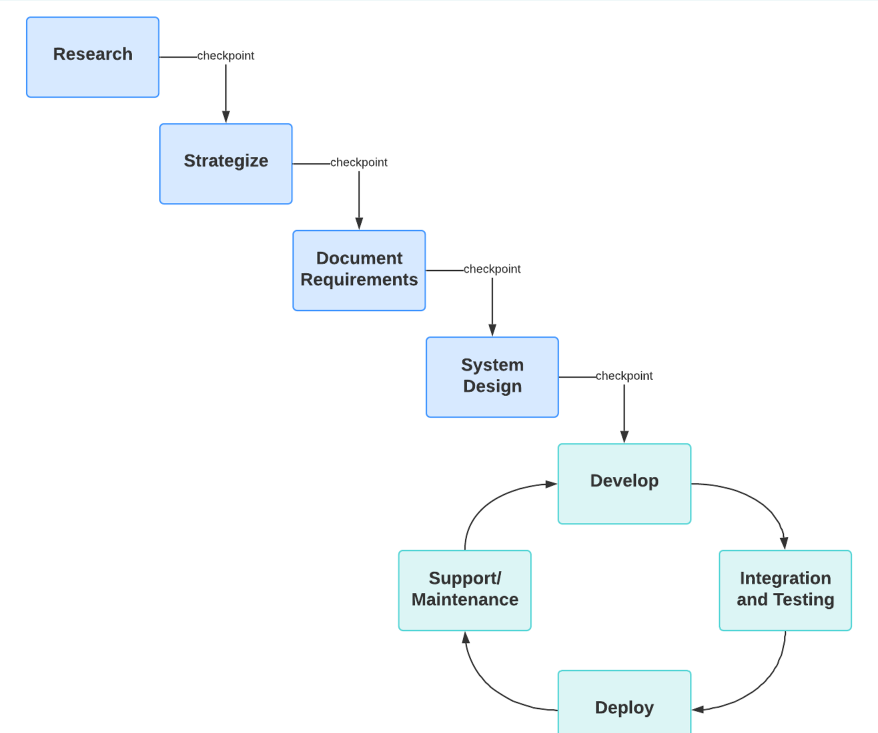


System Architecture & Techs Used



Project Management

- **Waterfall + Agile hybrid**
- Takes advantage of both models
- **Development and Implementation: Agile**
  - Divide the whole stage into many small sprints and iteratively complete them
  - More flexible time management
- **Other stages: Waterfall**
  - Project requirement is fixed in early stage, no need to do iteratively
  - Saves overhead time



Source: Lucidchart

Project Timeline

Schedule	Tasks
2024/08-2024/09	Identifying Requirements and Planning: l Defining project topic, defining functional requirements and rough design of overall system architecture l Preparation of Detail Project Plan
2024/10	Designing: l Process flows/System flows diagrams l UI design in Figma
2024/11-2024/12	Development and Implementation: l Frontend: completed prototype for most pages of the web app client and most major features not involving backend Interim Report and Presentation
2025/01-2025/02	Development and Implementation: l Backend: main web server
2025/03-2025/04	Development and Implementation l Backend: recommendation engine l Frontend: completed the remaining features involving the backend web server or the recommendation engine, e.g. fetch of recommended product lists Final Report, Presentation, Poster and Video

Summary

1st Sem: Define project topic and requirements, planning, designing, FE implementation

2nd Sem: BE implementation, integration of FE and BE, project deliverables

03

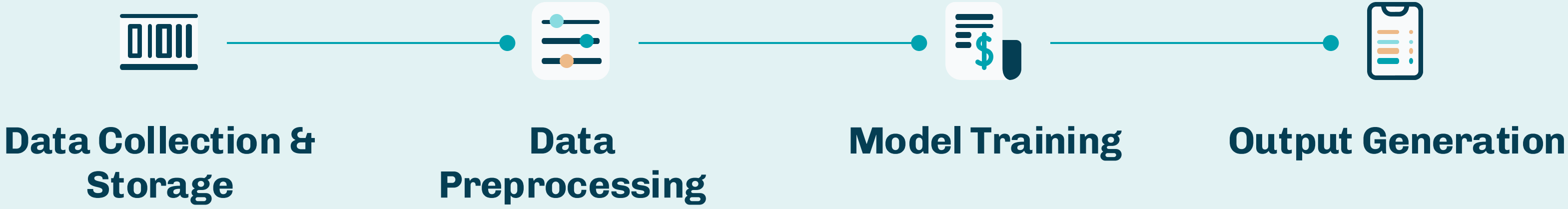
Project Results

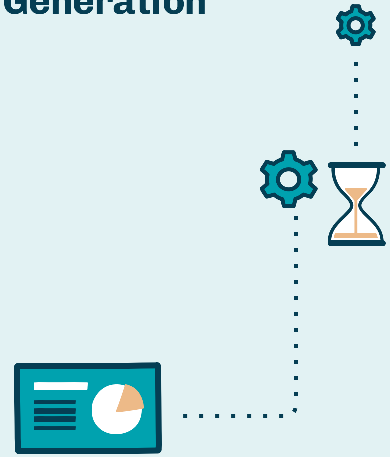
Web App Demo

Web App - Features Summary

- User account management + Authentication:
- Search products by name
- View products by category
- Product Sections: Personalized Recommendation + Hot items
- Product page: Product details, ordering, bookmarking, review
- Shopping Cart and Bookmarks
- Checkout
- Seller's Dashboard: View statistics, launch/manage products, manage received orders

Recommendation Engine - Workflow





Recommendation Engine - Workflow

Data Collection & Storage

- **Tracks user interactions** with web app client, e.g., viewing/purchasing/reviewing/bookmarking a product, clicking a recommended product
- Each user interaction is **stored in database**

Data Preprocessing

- **Fetch raw data from the database:** data frames of user/product/review/events
- **Clean data**: e.g., handling missing/null/extreme values
- **Extract part of the data/transform data**, so it can be received by the models as input

Model Training

- **Models used: Content-based Filtering, Collaborative Filtering, Hybrid**
- **Content-based:** recommends products with similar properties as products the user previously interacted with
- **Collaborative**: recommend products interacted by similar users

Output Generation

- **Each product is given a score, which decides its ranking** on the recommended product list. Products ranked top n will be displayed to the user.
- The scoring method depends on the model

Recommendation Engine – Data Collection

Collection:

- **Tracks user interactions** with web app client, e.g. viewing a product, purchasing a product, adding a review, clicking a recommended product, adding a product to bookmarks (using JWT and cookie)
- Each user interaction is a **'user event'**

Storage:

- Stored in user_event table in the DB (Frontend -> Web server -> DB through APIs)


```
_id: ObjectId('68057badac660961e31adee1')
event_type : "recommendation_click"
user_id : "67f60958d2d3797d255a226c"
product_id : "67e38a4c1a1f999cc90a0639"
timestamp : 2025-04-21T06:56:45.417+00:00
▸ data : Object

_id: ObjectId('68057badac660961e31adee5')
event_type : "product_view"
user_id : "67f60958d2d3797d255a226c"
product_id : "67e38a4c1a1f999cc90a0639"
timestamp : 2025-04-21T06:56:45.557+00:00
▸ data : Object
```

Recommendation Engine – Data Preprocessing

- **Fetch raw data from DB:** dataframes of user/product/review/events
- **Clean data**: e,g, handling missing/null/extreme values
- **Extract part of data/transform data**: so it can be received by the models as input

_id: ObjectId('68057badac660961e31adee1')
event_type : "recommendation_click"
user_id : "67f60958d2d3797d255a226c"
product_id : "67e38a4c1a1f999cc90a0639"
timestamp : 2025-04-21T06:56:45.417+00:00
data : Object

_id: ObjectId('68057badac660961e31adee5')
event_type : "product_view"
user_id : "67f60958d2d3797d255a226c"
product_id : "67e38a4c1a1f999cc90a0639"
timestamp : 2025-04-21T06:56:45.557+00:00
data : Object

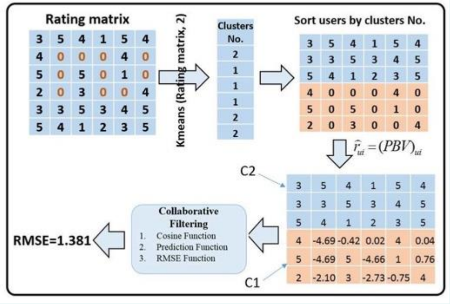
Recommendation Engine – Model Training

- **Models used: Content-based Filtering, Collaborative Filtering, Hybrid**
- **Content-based**
    - Characteristic: recommends products with similar properties as products the user previously interacted with (focus on item properties)
    - Receive product dataframes containing product metadata (descriptions, categories, features)
    - feature extraction by TF-IDF vectorization (find a term's importance in a text by calculating no. of times the term appears in the text and total no. of terms in the text)
    - Compute **each product's similarity to every other product** by Cosine Similarity

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Recommendation Engine – Model Training

- **Models used: Content-based Filtering, Collaborative Filtering, Hybrid**
- **Collaborative**
  - Characteristic: recommend products interacted by similar users (focus on user behaviour patterns)
  - Receive product reviews dataframes (contains user id, product id and rating, forming a user-item interaction matrix)
  - Decompose the matrix with SVD algorithm to find out user latent vectors (represents user preferences across latent dimensions), item latent vectors (represents product characteristics across same dimensions) and bias terms (Global average, user bias, and item bias)
  - Compute a **predicted rating for each product** based on these terms
- **Hybrid**: combines a weighted output from both content-based and collaborative filtering



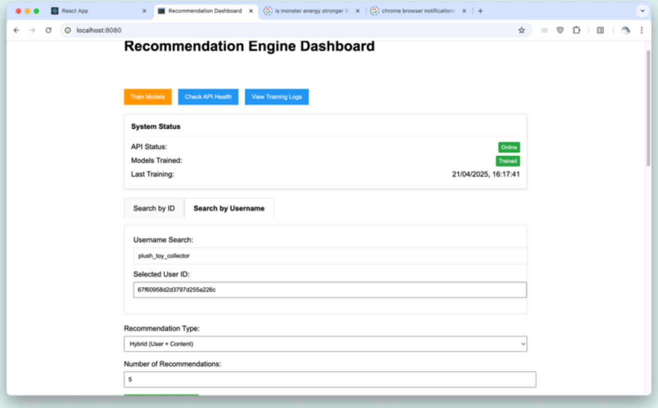
Recommendation Engine – Output Generation

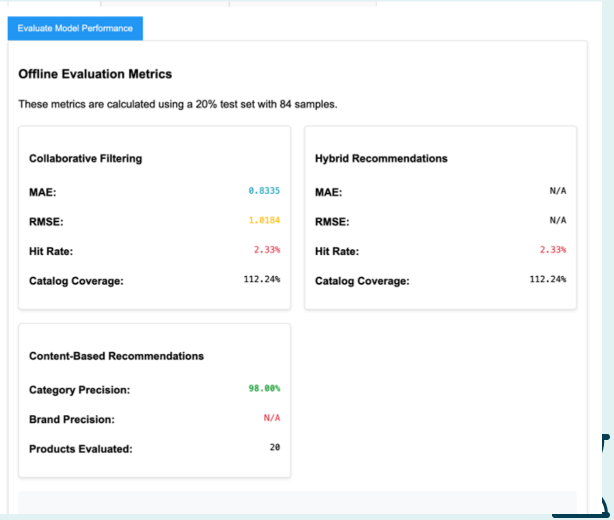
- After the model training, **each product is given a score which decides its ranking** on the recommended product list. Products ranked top n will then be sent to the Frontend and displayed to the user. The scoring method depends on the model:
  - Content-based: **the product's aggregated similarity to each other product,** calculated by Cosine Similarity
  - Collaborative: **the product's predicted rating by a specific user**, calculated by SVD algorithm
  - Hybrid: Content-based score * Content-based weighting + Collaborative score * Collaborative weighting

Testing and Evaluation

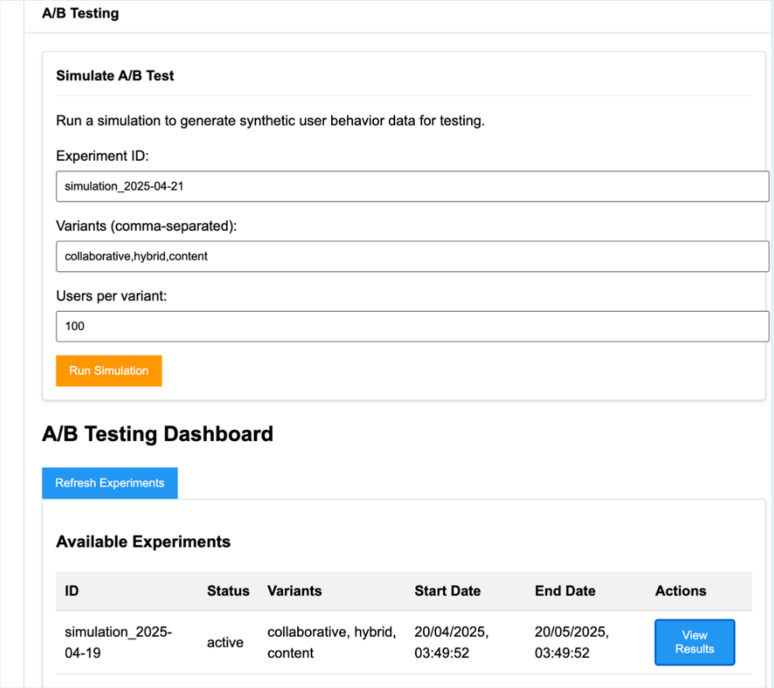
- **Offline testing**:
    - **MAE (Mean Absolute Error), RMSE (Root Mean Square Error):** measure difference between Collaborative Filtering's predicted rating and actual rating. The lower the better
    - **Hit Rate**: The proportion of users who received at least one relevant product recommendation among the list of recommended products they receive. The higher the better, but highly dependent on by size of the recommendation list
- **Online A/B testing (Offline simulation)**
    - Create some synthetic users (as there are no actual users yet)
    - Just like real-life A/B testing, divide users into 2 or more groups, each group use different recommendation model
    - **Predict each group's users' feedback** to recommendations by probabilistic assumptions (e.g. probability for a user to click on a recommended product = ?)
    - Observe which group has better predicted feedback -> the model for that group has better performance
        - Feedback metrics: Click-through rate, Conversion rate, purchase per users, etc.

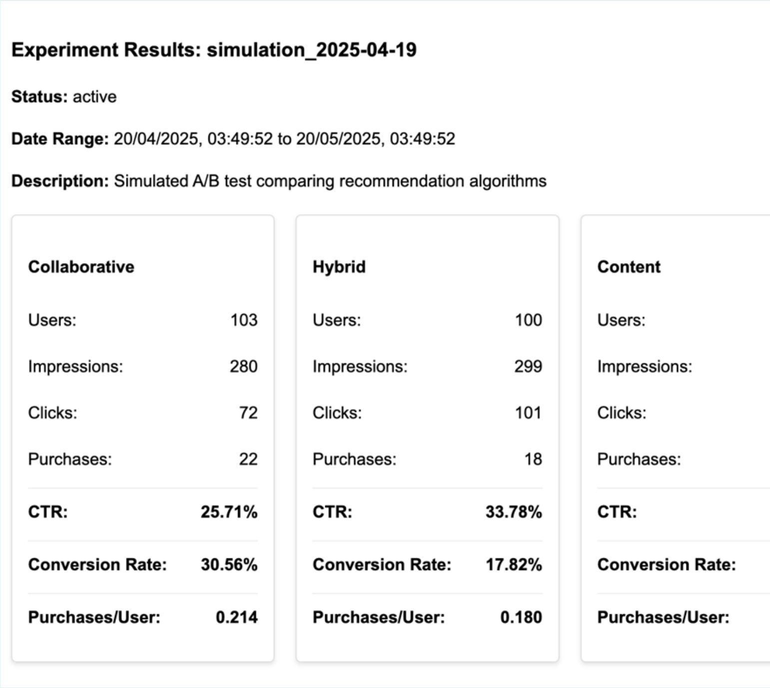
Recommendation System Dashboard





Recommendation System Dashboard





04

Future Works

Future Developments

The project is only implemented to a limited extent and has not reached production level, hence it has a large potential for future developments. For example:

- **Deployment on cloud host**: allow the implementation of features that require multiple actual users' feedback or interaction, such as the real-life A/B testing for the recommendation engine mentioned previously or real-time chatting between buyer and seller/customer service, or other production-level features.
- **Security**: Implement production-level security features like improved user authentication, API security, and server security
- **Infrastructure**: Streamline project deployment by containerization (Docker) and CI/CD Pipeline
- **UI/UX**: Implement mobile app version for the web app/implement responsive design to existing web app
- **Recommendation engine**: implement more extensive and online testing, import external datasets to improve cold start problem

05

Q&A

References

Banerjee, P. (2020). Recommender Systems in Python. Retreived from: https://www.kaggle.com/code/prashant111/recommender-systems-in-python

Deutschman, Z., (2023). Recommender Systems: Machine Learning Metrics and Business Metrics. Retrieved from: https://neptune.ai/blog/recommender-systems-metrics

EVIDENTLY AI (2025). 10 metrics to evaluate recommender and ranking systems. Retreived from: https://www.evidentlyai.com/ranking-metrics/evaluating-recommender-systems

GeeksforGeeks (2024). SVD in Recommendation Systems. Retrieved from: https://www.geeksforgeeks.org/svd-in-recommendation-systems/

GeeksforGeeks (2025). Understanding TF-IDF (Term Frequency-Inverse Document Frequency). Retreived from: https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/

HK01 (2022). 高達模型炒風不斷｜MG大魔$300炒到$900　轉售Gundam模型月入10萬原文網址: 高達模型炒風不斷｜MG大魔$300炒到$900　轉售Gundam模型月入10萬 | 香港01. Retrieved from: https://www.hk01.com/%E9%81%8A%E6%88%B2%E5%8B%95%E6%BC%AB/775830/%E9%AB%98%E9%81%94%E6%A8%A1%E5%9E%8B%E7%82%92%E9%A2%A8%E4%B8%8D%E6%96%B7-mg%E5%A4%A7%E9%AD%94-300%E7%82%92%E5%88%B0-900-%E8%BD%89%E5%94%AEgundam%E6%A8%A1%E5%9E%8B%E6%9C%88%E5%85%A510%E8%90%AC

HK Ulifestyle (2024). Retrieved from: https://www.facebook.com/photo.php?fbid=911125491054751&id=100064719244865&set=a.631855555648414

Hug (2015). Welcome to Surprise' documentation! Retrieved from: https://surprise.readthedocs.io/en/stable/

Kumar, S., (2024). A Guide to User Behavior Modeling. Retrieved from: https://blog.reachsumit.com/posts/2024/01/user-behavior-modeling-recsys/

Milankovich, M. (2015). The Cold Start Problem for Recommender Systems. Medium. Retrieved from: https://medium.com/@markmilankovich/the-cold-start-problem-for-recommender-systems-89a76505a7

Scikit-learn (2025). Scikit-learn. Retrieved from: https://scikit-learn.org/stable/index.html

Shaw, A. (2019). Product Recommendation System for e-commerce. Kaggle. Retrieved from: https://www.kaggle.com/code/shawamar/product-recommendation-system-for-e-commerce

World Integrated Trade Solution (2019). Toys nes imports by country in 2019. Retrieved from: https://wits.worldbank.org/trade/comtrade/en/country/ALL/year/2019/tradeflow/Imports/partner/WLD/product/950390