CONTEXTUAL RECOMMENDATION SYSTEM

IDEA BRIEF:

This recommendation engine is specifically for an ecosystem where multiple apps belongs to same system and we can get all possible type of data for a specific user to personalize it.
On each request from any app, system will produce list of the things which can be done by the user on the basis of his past behaviour with the help of the data from all of the apps.
Example:

1. User booked a cab from place-1 to place-2 using go-ride/go-car and we have data of his food orders what he usually orders and likes so this engine will recommend some good places near to place-2 to get similar food
2. User is using go-mart app to order some grocery or go-massage app then system will show notifications if you would like to order some snacks from go-food as well
3. You booked a ride to somewhere then it can show recommendation from go-tix for booking nearby shows and events based on user's previous bookings

So this way with the multiple type of dataset user recommendations can be highly personalised which can subsequently lead to higher number of uses of a particular app with the support of others.

TECHNOLOGY:

1. **Language:** Python (Good support for ML and recommendation algorithms)
2. **Database:** MongoDB
3. **Platform**: Android

APPROACH: For recommendation system we seeks to predict the preference that a user would give to an item.

Challenges :

1. **Scalability** : We are going to deal with data from multiple apps. Thus execution speed critical as the size of the data will be huge. Fast and efficient algorithm is required.
2. **Sparsity** : As we are dealing with multiple apps in a ecosystem, even the most active users will only have rated(used/bought) a small subset of the overall database. Thus, even the most popular items will have very few ratings.

We will be using **XGBoost** algorithm to solve the problem. We chose this algorithm because :

1. Execution Speed
    - **Parallelization** all of CPU cores during training.
    - **Distributed Computing** for training very large models using a cluster of machines.
    - **Out-of-Core Computing** for very large datasets that don't fit into memory.
    - **Cache Optimization** of data structures and algorithm to make best use of hardware.
2. Model Performance
    - XGBoost dominates classification and regression predictive modeling problems.
    - **Continued Training** so that you can further boost an already fitted model on new data.

TEAM ROLE:

Divya : Pipeline setup, Android integration, database handling and scaling,
Sonika : Data Cleaning, feature engineering and model training choosing the suitable parameters.