SYSTEM OVERVIEW: FASHION RECOMMENDATION SERVICE

OVERVIEW

The fashion recommendation system is designed to provide personalized outfit suggestions to users based on their preferences and past purchasing history, leveraging a robust backend architecture built on a Spring Boot framework. The system uses a combination of user demographics, inventory availability, and user-defined preferences to generate suitable fashion recommendations.

Architecture Components

Recommendation Controller: Handles HTTP requests from the user interface. It processes POST requests to the **/recommendations/{userId}** endpoint to initiate the outfit recommendation process.

Recommendation Service: The core business logic layer that processes the recommendation algorithm. It communicates with various services to fetch necessary data.

Services Layer: Consists of the Inventory Service, Purchase History Service, and Demographic Service, each responsible for fetching specific types of data from the database:

- **Inventory Service**: Retrieves available inventory items from the inventory table.
- **Purchase History Service**: Fetches a user's past purchase history from the purchase_history table.
- **Demographic Service**: Analyzes demographic data from the demographics table.

Database: A relational database management system that stores various tables such as users, inventory, purchase_history, and demographics.

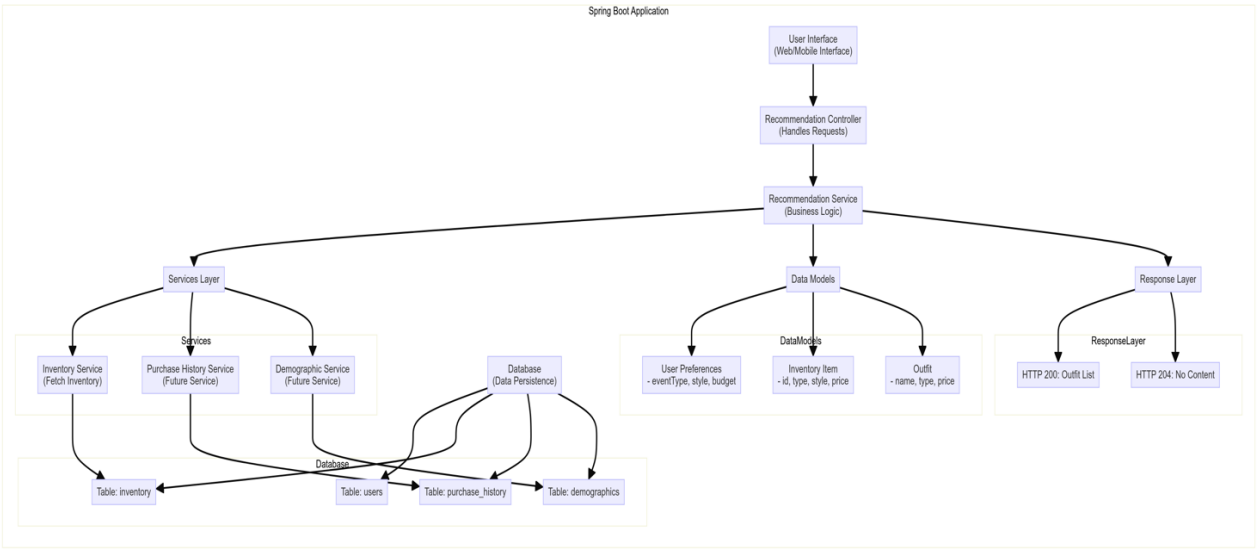
Data Models: The database contains several tables related to users, inventory items, and outfits, defined as follows:

- **User**: Identified by userId, includes demographic information.
- **UserPreferences**: Stores preferences for event type, style, and budget linked to the user.
- **InventoryItem**: Details of each inventory item including ID, type, style, and price.
- **Outfit**: Comprises items that form a complete outfit.
- **PurchaseHistory**: Contains historical purchase data linked to a user.
- **Feedback**: Allows users to leave feedback on outfits.

Interaction Flow

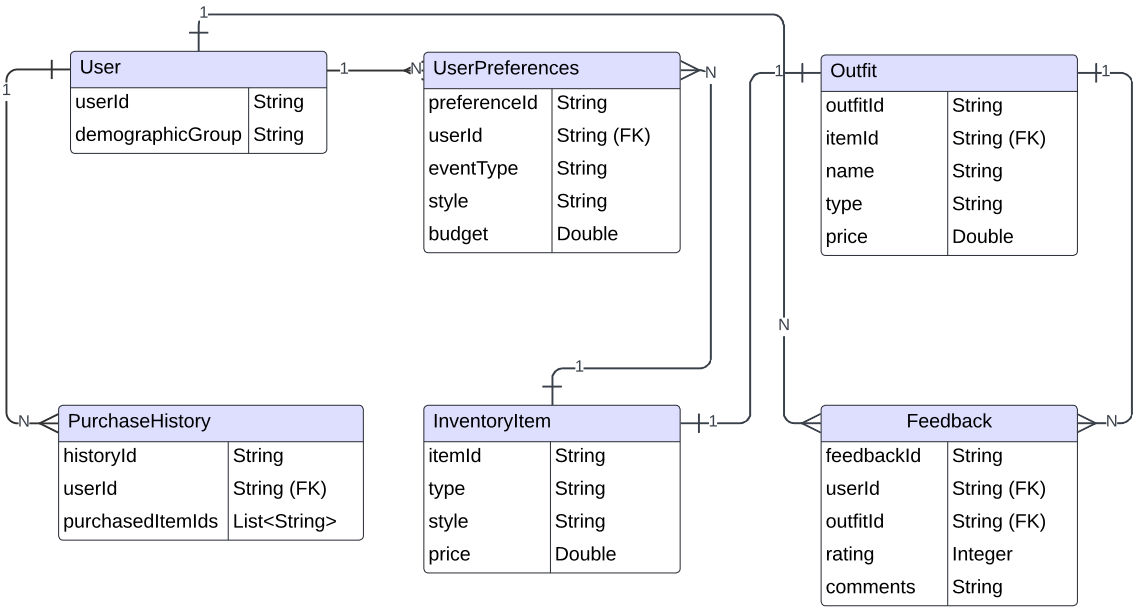
1. **User to Recommendation Controller**: The user sends a POST request to /recommendations/{userId}.
2. **Recommendation Controller to Recommendation Service**: Transfers the request with userId and user preferences.
3. **Recommendation Service Interactions**:
    - Calls Inventory Service to get current inventory.
    - Optionally queries Purchase History and Demographic Services for additional user data.
4. **Data Retrieval**: Each service fetches relevant data from the database and returns it to the Recommendation Service.
5. **Recommendation Compilation**: The Recommendation Service processes all data to formulate outfit recommendations.
6. **Response to User**: The Recommendation Controller returns either a list of outfits (HTTP 200) or a no-content status (HTTP 204) if no suitable outfits are found.

SYSTEM ARCHITECTURE DIAGRAM



DATABASE ENTITY RELATIONSHIP DIAGRAM

Entity Relation Diagram



P.S. Coding Task Focus

The coding task solution primarily targeted the development of key components within the backend services of the tool. Although the complete design integrates multiple services for a fully functional system, the coding work was specifically directed at developing one particular endpoint as required for the task.