Technical Document for DatingMatch Backend

Project Overview

Project Name: DatingMatch

Description: A backend program designed for matchmaking, facilitating user compatibility checks based on shared attributes and preferences. This API supports user registration and compatibility calculations using a set-based matching algorithm.

Technologies Used

- Backend Framework: Spring Boot
- Database: MongoDB
- Algorithm: Set-Based Matching Algorithm

API Endpoints

1. Welcome Endpoint

Method: GET Endpoint: /
Description: Displays a welcome message.
Response: Welcome to the Dating App Matchmaking API

2. Save User Endpoint

Method: POST Endpoint: /api/v1/match/save

Description: Saves a new user to the database.

Request Body Example: {

```
  "name": "string",
  "age": 25,
  "gender": "string",
  "interested_in": "string",
  "location": "string",
  "hobbies": ["string1", "string2"],
  "interests": ["string1", "string2"],
  "occupation": "string",
  "education_level": "string",
  "personality_traits": ["string1", "string2"]
```

}

3. Match One User with All Users

Method: POST Endpoint: /api/v1/match/{user}

Description: Matches a specific user with all other users in the database.

Path Variable: user (String): ID of the user to be matched.

4. Match Two Users

Method: POST Endpoint: /api/v1/compatibility/{user1}/{user2}

Description: Calculates compatibility between two selected users.

Path Variables:

- user1 (String): ID of the first user.
- user2 (String): ID of the second user.

Database Structure

User Collection

Field	Type	Description
id	String	Unique identifier for the user.
name	String	Name of the user.
age	Integer	Age of the user.
gender	String	Gender of the user.
interested_in	String	Preferred gender for matching.
location	String	User's location.
hobbies	List[String]	List of hobbies.
interests	List[String]	List of interests.
occupation	String	Occupation of the user.
education_level	String	Education level of the user.
personality_traits	List[String]	Personality traits of the user.

Matching Algorithm

Set-Based Matching Algorithm

1. Input: Two users with their respective attributes.
2. Steps:

- Calculate intersection and union of set-based attributes (e.g., hobbies, interests, personality traits).
- Compute compatibility score as:
- Compatibility Score = Intersection Size / Union Size
- Additional weight is added for matching attributes like location, age, occupation, and education level.

3. Output: Compatibility score and shared attributes.

Sample Responses

Save User

POST Request: /api/v1/match/save

Response:


```
{
  "message": "User saved successfully",
  "userId": "user123"
}
```


Match One User with All

POST Request: /api/v1/match/user123

Response:


```
[
  {
    "matchedUserId": "user456",
    "compatibilityScore": 0.75,
    "sharedAttributes": {
      "hobbies": ["reading"],
      "interests": ["traveling"]
    }
  },
  {
    "matchedUserId": "user789",
    "compatibilityScore": 0.60,
    "sharedAttributes": {
      "hobbies": ["music"],
      "interests": ["sports"]
    }
  }
]
```

Match Two Users

POST Request: /api/v1/compatibility/user123/user456

Response:


```
{
   "user1_id": "user123",
   "user2_id": "user456",
   "compatibility_score": 0.80,
   "sharedAttributes": {
      "hobbies": ["music", "reading"],
      "interests": ["traveling"]
   }
}
```