Summary Document: RESTful API User Search

Objective:

Create a RESTful API with a linked database to search for users based on specific query parameters.

Requirements:

1. The API should have an endpoint **`/api/users`** with a mandatory query parameter: **`first_name`**.
2. The API should search the user table for all users with the beginning of **`first_name`** matching the provided value.
3. If matching users are found, return the list of matching users in a JSON response.
4. If no matching users are found, call an external API **`https://dummyjson.com/users/search?q=first_name`** with the **`first_name`** parameter, save the resulting users to the user table, and return them in the response.

Solution:

1.Import necessary libraries:

- **`requests`** for making HTTP requests to the external API.
- **`JsonResponse`** and **`HttpResponseBadRequest`** from **`django.http`** for handling HTTP responses.

2.Define the **`user_search`** function that handles the **`/api/users`** endpoint.

- Retrieve the query parameters from the request (**`id`**, **`first_name`**, **`last_name`**, **`age`**, **`gender`**, **`email`**, **`phone`**, **`birth_date`**).
- Validate that at least one of the parameters is provided; otherwise, return a **`400 Bad Request`** response.
- Create an empty **`filters`** object of type **`Q`** to hold the filter conditions.

3.Apply the filters to the **`matching_users`** queryset:

- If a query parameter is provided, add the corresponding filter condition to the **`filters`** object.
- Use the **`Q`** objects in combination with the **`&=`** operator to combine the filter conditions with the **`AND`** operator.

4.Perform the search:

- Filter the **`User`** objects using the **`filters`** object, retrieving all the matching users in a single query.

5.Handle the search results:

- If matching users are found:
- Create a list of dictionaries containing the user data (id, first_name, last_name, age, gender, email, phone, birth_date).
- Convert the **`birth_date`** to the desired format (**`%Y-%m-%d`**) using **`strftime`**.
- If no matching users are found:
- Call the external API **`https://dummyjson.com/users/search?q=first_name`** with the provided **`first_name`** parameter.
- Extract the user data from the response and assign it to **`users_data`**.
- Create new user objects in the database using the extracted user data and store their details in the **`new_users`** list.

6.Return the response:

- If matching users were found, return a JSON response containing the **`users_data`** list.
- If no matching users were found, return a JSON response containing the **`new_users`** list.

Conclusion:

The implemented code provides a RESTful API with an endpoint to search for users based on query parameters. It filters the users based on the provided parameters and returns the matching users in a JSON response. If no matching users are found, it calls an external API, saves the retrieved users to the database, and returns them in the response.