FastAPI CRUD Operations Assignment

Objective: Build a simple API that manages a list of users. The users will have the following fields:

- id (int): Unique identifier for each user.
- name (str): Name of the user.
- phone_no (str): Phone number of the user.
- address (str): Address of the user.

You are required to implement the following operations:

Endpoints to implement:

1. Create a new user:
   - POST /users/
   - Request Body:

```
{
  "id": 1,
  "name": "John Doe",
  "phone_no": "1234567890",
  "address": "123 Main St"
}
```

   - Response: Status code 201 with a confirmation message.

2. Read user by id:
   - GET /users/{id}
   - Path parameter: id (int)

- Response:
  - If the user exists, return the user details in JSON.
  - If not, return a 404 status code with a "User not found" message.

3. Read users by name:
- GET /users/search?name={name}
- Query parameter: name (str)
- Response:
  - If users with the provided name exist, return a list of users.
  - If no users found, return an empty list.

4. Update user details:
- PUT /users/{id}
- Path parameter: id (int)
- Request Body:

```
{
  "name": "John Updated",
  "phone_no": "9876543210",
  "address": "456 Updated St"
}
```

- Response: Status code 200 with a message confirming the update.

5. Delete user by id:
- DELETE /users/{id}
- Path parameter: id (int)
- Response:
  - If the user exists, return a message confirming the deletion.

- If not, return a 404 status code with a "User not found" message.

Hints:

- You will need to create an in-memory storage for users, such as a list or dictionary.
- Use pydantic models to validate request bodies.
- Implement appropriate HTTP status codes and responses.

Example API Interaction:

1. Create a new user:
   - Request:

     POST /users/

     {

       "id": 1,

       "name": "John Doe",

       "phone_no": "1234567890",

       "address": "123 Main St"

     }
   - Response:

     {

       "message": "User created successfully"

     }

2. Get user by ID:
   - Request:

     GET /users/1
   - Response:

{

"id": 1,

"name": "John Doe",

"phone_no": "1234567890",

"address": "123 Main St"

}

3. Search users by name:

- Request:

GET /users/search?name=John

- Response:

[

{

"id": 1,

"name": "John Doe",

"phone_no": "1234567890",

"address": "123 Main St"

}

]

4. Update user details:

- Request:

PUT /users/1

{

"name": "John Updated",

"phone_no": "9876543210",

"address": "456 Updated St"

}

- Response:

{

"message": "User updated successfully"

}

5. Delete user by ID:

- Request:

DELETE /users/1

- Response:

{

"message": "User deleted successfully"

}

Requirements:

- Install FastAPI and Uvicorn:

pip install fastapi uvicorn

- Use pydantic for data validation.

- Run the app with Uvicorn:

uvicorn main:app --reload