**User Management Microservice Documentation**

**Overview**

The User Management Microservice provides RESTful API endpoints for managing user information, including user creation, modification, deletion, and authentication. The microservice is designed with object-oriented principles and uses an in-memory data store for simplicity.

**Base URL**

The base URL for the microservice is <http://localhost:8082>.

**API Endpoints**

1. Add User

* Endpoint: POST /users/add
* Description: Adds a new user to the system.
* Request Payload:

{

"name":"Govinda",

"roles":"Developer",

"password":"Govinda@123"

}

**Response**:

* Success: Returns the success message.
* Failure: Returns an error message.

2. Edit User

* Endpoint: PUT /users/{ID}
* Description: Edits an existing user in the system.
* Request Payload:

{

"name":"Govinda",

"roles":"Developer",

"password":"Govinda@1234"

}

**Response**:

* Success: Returns the success message.
* Failure: Returns an error message.

3. Delete User

* Endpoint: DELETE /users/{ID}
* Description: Deletes an existing user in the system.
* Path Variable:

userId (String): The unique identifier of the user to be deleted.

**Response**:

* Success: Returns the success message.
* Failure: Returns an error message.

4. Authenticate User

* Endpoint: POST /users/authenticate
* Description: Authenticates a user based on the provided credentials.
* Request Parameters:

username (String): The username of the user.

password (String): The password of the user.

Response:

* Success: Returns "Authentication successful."
* Failure: Returns "Authentication failed."

**Password Validation**

The microservice enforces the following password complexity criteria during user creation and modification:

* At least 8 characters.
* At least one digit.
* At least one lowercase letter.
* At least one uppercase letter.
* At least one special character (e.g., @, #, $, etc.).
* No whitespaces allowed.

**Username Validation**

* If Username already exists in database then user not allowed to add (needs to try with new one).
* If User does not exists then that user not able to update or delete records.

**Unit Tests**

Unit tests are provided for major services and functions in the UserServiceTest class. These tests cover user addition, editing, and deletion functionalities.