## **LOGIN SYSTEM API DOCUMENT**

## 

## **1. User Signup**

## 

## **Endpoint:** /user/signup

## **Method:** POST

## **Description:** Registers a new user.

## **Headers:**

## **applicationid (string, required):** Your application ID

## **clientid (string, required):** Your client ID

## **Body:**

## **fullname (string, required):** Full name of the user.

## **username (string, required):** Desired username.

## **password (string, required):** User password.

## **email (string, required):** User email address.

## **phone\_number (string, required):** User phone number.

## **Response:**

## **202 Accepted:** If the verification link has been sent successfully.

## **400 Bad Request:** If there are missing fields or validation fails.

## **500 Internal Server Error:** If there's a server-side issue.

## 

## **2. Verify Email**

## 

## **Endpoint:** /user/verify\_email/<token>

## **Method:** GET

## **Description:** Verifies user's email using the verification token.

## **Path Parameters:**

## **token (string, required):** Verification token received via email.

## **Response:**

## **201 Created:** If email verified and user created successfully.

## **400 Bad Request:** If the token is invalid or expired.

## 

## **3. User Signin**

## 

## **Endpoint:** /user/signin

## **Method:** POST

## **Description:** Allows users to log in.

## **Body:**

## **username (string, required):** User's username.

## **password (string, required):** User's password.

## **Response:**

## **200 OK:** If login successful. Returns a JWT token.

## **400 Bad Request:** If username or password is invalid.

## 

## **4. Forgot Password**

## 

## **Endpoint:** /user/forgot\_password

## **Method: POST**

## **Description:** Sends a password reset link to the user's email.

## **Body:**

## **email (string, required):** User's email address.

## **Response:**

## **200 OK:** If the password reset link has been sent successfully.

## **400 Bad Request:** If the email is invalid or missing.

## **404 Not Found:** If the user is not found.

## 

## **5. Reset Password Page**

## **Endpoint:** /user/reset\_password/<token>

## **Method:** GET

## **Description:** Renders the reset password page.

## **Path Parameters:**

## **token (string, required):** Reset password token received via email.

## **Response:**

## Renders a HTML page.

## 

## **6. Update Password**

## **Endpoint:** /user/update\_password

## **Method:** POST

## **Description:** Updates user's password.

## **Body:**

## **token (string, required):** Reset password token.

## **password (string, required):** New password.

## **confirm\_password (string, required):** Confirm new password.

## **Response:**

## **200 OK:** If password updated successfully.

## **400 Bad Request:** If token is invalid or passwords do not match.

## 

## **7. Upload Image**

## **Endpoint:** /user/upload\_image

## **Method:** POST

## **Description:** user uploads an image.

## **Headers:**

## **Authorization:** token obtained during login.

## **Form Data:**

## **file (file, required):** Image file to upload.

## **Response:**

## **201 Created:** If image uploaded successfully.

## **400 Bad Request:** If no file found or file format not supported.

## **8. Change Password**

## **Endpoint:** /user/change\_password

## **Method:** POST

## **Description:** Changes user's password.

## **Headers:**

## **Authorization:** token obtained during login.

## **Body:**

## **old\_password (string, required):** Current password.

## **new\_password (string, required):** New password.

## **confirm\_password (string, required):** Confirm new password.

## **Response:**

## **200 OK:** If password changed successfully.

## **400 Bad Request:** If old password is incorrect or passwords do not match.

## **Note:**

## Ensure to pass necessary headers, body, and tokens as mentioned in the documentation for each endpoint.

## For file uploads, use the multipart/form-data format.

## For authentication, include the JWT token in the Authorization header.