

API Documentation

1. Register User

- **URL:** `/register`
- **Method:** `POST`
- **Description:** Registers a new user with the provided details.
- **Request Body:**
 - `UserName` (String, required): Unique username for the user.
 - `Name` (String, required): Full name of the user.
 - `Password` (String, required): User's password.
 - `CPassword` (String, required): Confirmation of the user's password.
 - `PhoneNo` (String, required): User's phone number.
 - `EmailID` (String, required): User's email address.
- **Response:**
 - `message`: Success message indicating successful registration.
- **Error Responses:**
 - `400 Bad Request`: If any provided data is invalid or missing.
 - `500 Internal Server Error`: If an unexpected error occurs during registration.

2. User Login

- **URL:** `/login`
- **Method:** `POST`
- **Description:** Logs in an existing user with the provided credentials.
- **Request Body:**
 - `UserName` (String): Username of the user.
 - `EmailID` (String): Email address of the user.
 - `Password` (String, required): User's password.
- **Response:**
 - `result`: Boolean indicating the login result.
 - `message`: Success message indicating successful login.
- **Error Responses:**
 - `400 Bad Request`: If credentials are invalid or missing.
 - `500 Internal Server Error`: If an unexpected error occurs during login.

3. Authenticate User

- **URL:** `/authenticate`
- **Method:** `GET`
- **Description:** Authenticates a user based on the provided JWT token.
- **Request Headers:**
 - `Authorization`: JWT token in the format `Bearer <token>`.

- ****Response:****
 - `success`: Boolean indicating the authentication result.
 - `user`: Object containing user details.
- ****Error Responses:****
 - `401 Unauthorized`: If authentication fails due to invalid or missing token.
 - `500 Internal Server Error`: If an unexpected error occurs during authentication.

2. Create Gig

- ****URL:**** `/Create/gigs`
- ****Method:**** `POST`
- ****Description:**** Creates a new gig with the provided details.
- ****Request Body:****
 - ****Form Data:**** Multipart form data with the following fields:
 - `data` (String, required): JSON string containing:
 - `title` (String, required): Title of the gig. Must be between 10 and 100 characters.
 - `cost` (Number, required): Cost of the gig. Minimum value is 100.
 - `description` (String, required): Description of the gig. Must be between 500 and 1500 characters.
 - `tags` (Array of Strings, required): Tags for the gig. Up to 10 tags, each between 1 and 50 characters.
 - `deliveryTime` (Number, required): Delivery time in hours. Minimum value is 1 hour.
 - `files` (Array of Files, required): Images for the gig. Minimum 1 image, maximum 5 images.
- ****Response:****
 - `message`: Success message indicating successful creation of the gig.
- ****Error Responses:****
 - `400 Bad Request`: If any provided data is invalid or missing. Specific error messages include:
 - "Title must be between 10 and 100 characters long."
 - "Description must be between 500 and 1500 characters long."
 - "A maximum of 10 tags are allowed."
 - "Each tag must be between 1 and 50 characters long."
 - "Please provide valid tags."
 - "The minimum cost of a gig is 100."
 - "The minimum delivery time for a gig is 1 hour."
 - "Only five images are allowed."
 - "At least 1 image is needed."
 - `500 Internal Server Error`: If an unexpected error occurs during gig creation.

3. Fetch Gigs

- ****URL:**** `/fetchGigs`
- ****Method:**** `GET`
- ****Description:**** Retrieves a list of gigs based on specified filters.
- ****Query Parameters:****
 - `title` (String, optional): Filters gigs by title (maximum 100 characters).

- `tags` (String, optional): Filters gigs by tags, separated by commas (maximum 50 characters per tag).
- `by` (String, optional): Filters gigs by username (maximum 20 characters).
- `gigId` (String, optional): Filters gigs by gig ID (maximum 35 characters).
- **Response:**
 - `success`: Boolean indicating the success of the request.
 - `data`: Array of gigs matching the specified filters.
- **Error Responses:**
 - `400 Bad Request`: If any provided data is invalid or missing.
 - `500 Internal Server Error`: If an unexpected error occurs during fetching.

5. Get Media

- **URL:** `/getmedia/:gid/:image`
- **Method:** `GET`
- **Description:** Retrieves media files associated with a gig.
- **URL Parameters:**
 - `gid` (String, required): ID of the gig.
 - `image` (String, required): Name of the media file.
- **Response:**
 - Media file if found.
- **Error Responses:**
 - `404 Not Found`: If the requested file is not found.
 - `500 Internal Server Error`: If an unexpected error occurs during file retrieval.