DU GAMES

This API is for du games all data operation api.

Admin

In API Admin side api operation work in this collection.

GET get all admin

localhost/DU_api/admins

GET /DU_api\admins

This endpoint is used to retrieve a list of administrators.

Request Body

This request does not require a request body.

Response

- Status: 200
- Content-Type: application/json

Response Body

The response contains a JSON object with the following keys:

- `verifyedUser`: An object containing verification information with keys `msg`, `data`, and `code`.
  - `msg`: A message related to user verification.
  - `data`: An object containing user data with keys `id`, `name`, `password`, `photo`, `email`, `iat`, and `exp`.
  - `code`: A verification code.
- `data`: An array of administrators with keys `admin_id`, `name`, `password`, `photo`, and `email_address`.
- `status_code`: A status code.
- `message`: A message related to the response.

AUTHORIZATION Bearer Token

Token {{jwt}}

GET getby id admin

localhost/DU_api/admin/2

This endpoint makes an HTTP GET request to retrieve information about the admin with ID 2. The response will be in JSON format with a status code of 200.

Request

No request body is required for this endpoint.

Response

The response will contain the following fields:

- `verifyedUser`: An object containing a `msg`, `data`, and `code` field.
  - `msg`: A message related to the verification process.
  - `data`: An object with fields like `id`, `name`, `password`, `photo`, `email`, `iat`, and `exp`.
  - `code`: A code related to the verification process.
- `status_code`: The status code of the response.
- `message`: A message related to the response.
- `data`: An object with fields like `admin_id`, `name`, `password`, `photo`, and `email_address`.

Please note that the actual data values are intentionally masked for privacy.

AUTHORIZATION Bearer Token

Token {{jwt}}

POST register admin

localhost/DU_api/admin

This endpoint allows administrators to perform actions on the DU system.

Request Body

- `name` (string, required): The name of the administrator.
- `password` (string, required): The password for the administrator.
- `photo` (string, optional): The photo of the administrator.

- `email` (string, required): The email address of the administrator.

Response

The response is in JSON format with the following schema:

json

```json
{
    "type": "object",
    "properties": {
        "message": {
            "type": "string"
        },
        "status_code": {
            "type": "integer"
        },
        "msg": {
            "type": "string"
```

Body raw (json)

json

```json
{
            "name": "meet",
            "password": "meet123",
            "photo": "meet.jpg",
            "email": "meet@gmail.com"
}
```

PUT edit user

localhost/DU_api/admin

Update Admin User

This endpoint allows updating the admin user details.

Request Body

- `id` (string): The ID of the admin user.
- `name` (string): The name of the admin user.
- `password` (string): The password of the admin user.
- `photo` (string): The photo of the admin user.

- `email` (string): The email of the admin user.

Response

- `verifyedUser` (object): An object containing verification details.
  - `msg` (string): Verification message.
  - `data` (object): Object containing updated admin user details.
    - `id` (number): The ID of the admin user.
    - `name` (string): The name of the admin user.
    - `password` (string): The password of the admin user.
    - `photo` (string): The photo of the admin user.
    - `email` (string): The email of the admin user.
    - `iat` (number): Issued at timestamp.
    - `exp` (number): Expiry timestamp.
  - `code` (number): Verification code.
- `message` (string): Additional message.
- `status_code` (number): Status code of the response.

AUTHORIZATION Bearer Token

Token	{{jwt}}

Body raw (json)

json

```
{
            "id": "3",
            "name": "kk",
            "password": "kk123",
            "photo": "kk.jpg",
            "email": "kk@gmail.com"
}
```

DELETE delete user

localhost/DU_api/admin/3

Delete Admin

This endpoint sends an HTTP DELETE request to localhost/DU_api/admin/3 to delete the admin with the ID 3.

Request Body

This request does not require a request body.

Response

The response will not contain a body, but will indicate the success or failure of the deletion operation.

AUTHORIZATION Bearer Token

Token	{{jwt}}

POST login

localhost/DU_api/login

Login API.

Log in Admin User in Two fildes to data enter

email and password.

Example data:

Email:nirav@gmail.com

Password:nirav123

add data in login api in body json formate

{ "email":"nirav@gmail.com", "password":"nirav123"}
like this

after login generate jwt tocken

example jwt beare tocken:

StartFragmenteyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MSwibmFtZSI6Im5pcmF2IGthZ2F0aGFyYSIsInBhc3N3b3
JkIjoibmlyYXYxMjMiLCJwaG90byI6Im5pcmF2LmpwZyIsImVtYWlsIjoibmlyYXZAZ21haWwuY29tIiwiaWF0IjoxNzIzNDMxNzU
1LCJleHAiOjE3MjM0MzUzNTV9.ZLS4ddkfFV0luSvnbfeGc58TRKJAiLR5MI9DNx90xHoEndFragment

AUTHORIZATION

Is Secret Base64Encoded	<is-secret-base64encoded>
Secret	{{jwt}}
Add Token To	<add-token-to>

Algorithm	<algorithm>
Payload	<payload>
Header Prefix	<header-prefix>
Query Param Key	<query-param-key>
Header	<header>

Body raw (json)

json

```json
{
    "email":"nirav@gmail.com",
    "password":"nirav123"
}
```

GET jwt verify

localhost/DU_api/token

AUTHORIZATION Bearer Token

Token	{{jwt}}

Feedback

in feedback table use this api.

POST add feed back

localhost/DU_api/feedback

Add Feedback

This endpoint allows the client to submit feedback by sending a POST request to the specified URL.

Request Body

- `name` (string, optional): The name of the person submitting the feedback.
- `category` (string, optional): The category to which the feedback belongs.
- `feedback` (string, required): The actual feedback content.

Response

The response will include the status of the feedback submission.

Body raw (json)

json

```
{
            "name": "jay patel",
            "category": "Car Chase",
            "feedback": "some will change in game"
}
```

GET get all feedback

localhost/DU_api/feedbacks

This endpoint retrieves feedback data from the server. The response is in JSON format and has the following structure:

json

```
{
    "verifyedUser": {
        "msg": "",
        "data": {
            "id": 0,
            "name": "",
            "password": "",
            "photo": "",
            "email": "",
            "iat": 0,
            "exp": 0
```

AUTHORIZATION Bearer Token

Token	{{jwt}}

GET getby id feed back

localhost/DU_api/feedback/3

The endpoint retrieves feedback for a specific user with the ID 3.

The response of this request can be documented as a JSON schema as follows:

json

```
{
    "type": "object",
    "properties": {
        "verifyedUser": {
            "type": "object",
            "properties": {
                "msg": {
                    "type": "string"
                },
                "data": {
                    "type": "object",
```

AUTHORIZATION Bearer Token

Token	{{jwt}}

PUT edit feedback

localhost/DU_api/feedback

The `PUT` request updates the feedback for a specific item via the `localhost/DU_api\feedback` endpoint.

Request Body

- The request body should be in raw JSON format and include the following parameters:
    - `id` (string): The ID of the item for which the feedback is being updated.
    - `name` (string): The name of the user providing the feedback.
    - `category` (string): The category of the item.
    - `feedback` (string): The feedback content.

Response

The response is in JSON format and follows the schema below:

json

```
{
  "type": "object",
  "properties": {
    "verifyedUser": {
      "type": "object",
      "properties": {
        "msg": { "type": "string" },
        "data": {
          "type": "object",
          "properties": {
```

AUTHORIZATION Bearer Token

Token	{{jwt}}

Body raw (json)

json

```
{
     "id": "5",
            "name": "olx patel",
            "category": "Car Chase",
            "feedback": "not good"
}
```

DELETE delete feed back

localhost/DU_api/feedback/5

DELETE /DU_api/feedback/5

This endpoint is used to delete a specific feedback entry with the ID of 5.

Request

No request body is required for this endpoint.

Response

The response for this request is a JSON object with the following schema:

json

```
{
  "message": "string"
}
```

- `message`: A string indicating the outcome of the delete operation.

AUTHORIZATION Bearer Token

Token	{{jwt}}

game developer

GET get all game

localhost/DU_api/games

The `GET` request to `localhost/DU_api\games` retrieves a list of game developer data.

Response

The response will be a JSON object with the following schema:

json

```
{
    "data": [
        {
            "gamedev_id": "",
            "gamedev_name": "",
            "gamedev_enrollment_no": "",
            "gamedev_game_name": "",
            "gamedev_photo": "",
            "gamedev_game_photo": "",
            "gamedev_linkdin_profile": "",
            "gamedev_gamefolder": ""
```

GET get by id

localhost/DU_api/game/2

Get Game Details

This endpoint retrieves the details of a specific game.

Request

Request URL

`GET localhost/DU_api/game/2`

Response

- Status: 401
- Content-Type: application/json

json

```
{
    "status_code": 0,
    "message": "",
    "data": {
        "gamedev_id": "",
        "gamedev_name": "",
        "gamedev_enrollment_no": "",
        "gamedev_game_name": "",
        "gamedev_photo": "",
        "gamedev_game_photo": "",
        "gamedev_linkdin_profile": "",
```

POST add game profile and game

localhost/DU_api/game

Endpoint Description

This endpoint is a POST request to localhost/DU_api\game. It is used to submit game-related information including name, enrollment, game name, photo, game photo, LinkedIn profile, and folder.

Request Body

- `name` (string): The name of the user.
- `enrollment` (string): The enrollment information of the user.
- `gameName` (string): The name of the game.
- `photo` (string): The user's photo.
- `gamePhoto` (string): The game's photo.
- `linkdinprofile` (string): The user's LinkedIn profile.
- `folder` (string): The folder information.

Response

The response is in JSON format with the following schema:

json

```
{
    "verifyedUser": {
        "msg": "",
        "data": {
            "id": 0,
            "name": "",
            "password": "",
            "photo": "",
            "email": "",
            "iat": 0,
            "exp": 0
```

- `verifyedUser` (object): Contains the verification status and user data.
    - `msg` (string): Verification message.
    - `data` (object): User data.
        - `id` (number): User ID.
        - `name` (string): User's name.
        - `password` (string): User's password.
        - `photo` (string): User's photo.
        - `email` (string): User's email.
        - `iat` (number): Issued at timestamp.
        - `exp` (number): Expiry timestamp.
    - `code` (number): Verification code.
- `message` (string): Additional message.
- `status_code` (number): Status code of the response.

AUTHORIZATION Bearer Token

Token	{{jwt}}

Body raw (json)

json

```
{
        "name": "maulik bhatt",
        "enrollment": "21010101111",
        "gameName": "Car Chase",
        "photo": "maulik.jpg",
        "gamePhoto": "chase.jpg",
        "linkdinprofile": "https://www.linkedin.com/in/maulikbhatt07/",
        "folder": "games/car-chase/index.html",
        "folder": "games/car chase/index.html"
}
```

PUT edit game profile or game

localhost/DU_api/game

Update Game Details

This endpoint allows the client to update game details by sending an HTTP PUT request to the specified URL.

Request Body

- `id` (integer): The ID of the game.
- `name` (string): The name of the game.
- `enrollment` (string): The enrollment details.
- `gameName` (string): The name of the game.
- `photo` (string): The photo of the game.
- `gamePhoto` (string): The photo of the game.
- `linkdinprofile` (string): The LinkedIn profile.
- `folder` (string): The folder details.

Response

The server responds with a status code of 200 and a JSON object containing the following fields:

- `verifyedUser` (object):
    - `msg` (string): A message related to the verification of the user.
    - `data` (object):
        - `id` (integer): The ID of the user.
        - `name` (string): The name of the user.
        - `password` (string): The password of the user.
        - `photo` (string): The photo of the user.
        - `email` (string): The email of the user.
        - `iat` (integer): Issued at timestamp.
        - `exp` (integer): Expiry timestamp.
    - `code` (integer): The verification code.
- `message` (string): Additional message from the server.
- `status_code` (integer): The status code of the response.

AUTHORIZATION Bearer Token

Token	{{jwt}}

Body raw (json)

json

```json
{
        "id":3,
        "name": "uet bhatt",
        "enrollment": "555",
        "gameName": "Car Chase",
        "photo": "maulik.jpg",
        "gamePhoto": "chase.jpg",
        "linkdinprofile": "https://www.linkedin.com/in/maulikbhatt07/",
        "folder": "games/car chase/index.html"
}
```

DELETE Delete game dev or game

localhost/DU_api/game/3

The endpoint sends an HTTP DELETE request to localhost/DU_api/game/3 to delete a specific game. Upon successful deletion, the response will follow a JSON schema.

AUTHORIZATION Bearer Token

Token	{{jwt}}

web developer

GET getall web developer

localhost/DU_api/webs

GET /DU_api/webs

This endpoint retrieves web developer information.

Request Body

This request does not require a request body.

Response

The response is in JSON format and has the following schema:

json

```
{
    "data": [
        {
            "webdev_id": "",
            "webdev_name": "",
            "webdev_role": "",
            "webdev_photo": "",
            "webdev_linkdin_profile": ""
        }
    ],
    "status_code": 0,
```

GET getby id web developer

localhost/DU_api/web/2

The endpoint localhost/DU_api/web/2 is an HTTP GET request that returns a JSON response. The last execution of this request returned a status code of 401 with the following JSON schema:

json

```
{
    "type": "object",
    "properties": {
        "status_code": {
            "type": "integer"
        },
        "message": {
            "type": "string"
        },
        "data": {
            "type": "object",
```

POST add web developer profile

localhost/DU_api/web

Request Description

This endpoint is used to make an HTTP POST request to localhost/DU_api\web. The request should include a raw request body with the following parameters:

- name (string): The name of the user.
- role (string): The role of the user.
- photo (string): The photo of the user.
- linkdin (string): The LinkedIn profile of the user.

Response

The response to this request is in JSON format with the following schema:

json

```json
{
    "verifyedUser": {
        "msg": "string",
        "data": {
            "id": 0,
            "name": "string",
            "password": "string",
            "photo": "string",
            "email": "string",
            "iat": 0,
            "exp": 0
```

- verifyedUser (object): An object containing user verification details.
  - msg (string): A message related to user verification.
  - data (object): An object containing user data.
    - id (number): The user ID.
    - name (string): The name of the user.
    - password (string): The user's password.
    - photo (string): The user's photo.
    - email (string): The user's email.
    - iat (number): The issued at timestamp.
    - exp (number): The expiration timestamp.
  - code (number): A code related to user verification.
- message (string): A general message related to the response.
- status_code (number): The status code of the response.

AUTHORIZATION Bearer Token

Token	{{jwt}}

Body raw (json)

json

```
{
        "name": "nirav kagathara",

        "role": "api manager",
        "photo": "nirav.jpg",
        "linkdin": "https://www.linkedin.com/in/nirav-kagathara-809781"
}
```

PUT edit game developer profile

localhost/DU_api/web

Update User API

This API endpoint is used to update user information.

Request Body

- **id** (number) - The unique identifier of the user.
- **name** (string) - The name of the user.
- **role** (string) - The role of the user.
- **photo** (string) - The URL of the user's photo.
- **linkdin** (string) - The LinkedIn profile URL of the user.

Response

- **verifyedUser** (object)
    - **msg** (string) - A message related to the verification of the user.
    - **data** (object)
        - **id** (number) - The unique identifier of the user.
        - **name** (string) - The name of the user.
        - **password** (string) - The password of the user.
        - **photo** (string) - The URL of the user's photo.
        - **email** (string) - The email address of the user.
        - **iat** (number) - The issued at timestamp.
        - **exp** (number) - The expiration timestamp.
    - **code** (number) - The status code related to user verification.
- **message** (string) - A general message related to the response.
- **status_code** (number) - The status code of the response.

AUTHORIZATION Bearer Token

Token	{{jwt}}

Body raw (json)

json

```json
{
        "id":3,
        "name": "subh kagathara",
        "role": "api manager",
        "photo": "nirav.jpg",
        "linkdin": "https://www.linkedin.com/in/nirav-kagathara-809781"
}
```

DELETE delete web devloper or web

localhost/DU_api/web/4

DELETE /DU_api/web/4

This endpoint is used to delete a specific resource identified by the ID "4".

Request Body

This request does not require a request body.

Response

The response for this request follows the JSON schema below:

json

```json
{
  "type": "object",
  "properties": {
    "message": {
      "type": "string"
    }
  }
}
```

The response contains a "message" property of type string, providing information about the outcome of the deletion operation.

AUTHORIZATION Bearer Token

Token	{{jwt}}