API Documentation

**Base URL**

<https://zaky.dikelasawan.my.id:8443/api>

﻿**1.Endpoint: User Register**

* **URL**: /register
* **Method**: POST
* **Description**: Registers a new user by validating the input data, saving the user's profile picture, hashing the password, and attaching a default role to the user.

**Request Headers**

* Accept : application/json

**Request Body**

The request body should be a JSON object with the following fields:

{

"name": "string",

"username": "string",

"email": "string",

"password": "string",

"birth\_date": "date",

"phone\_number": "integer",

"profile\_picture": "file",

}

**2. Endpoint: User Login**

* **URL**: /login
* **Method**: POST
* **Description**: Authenticates a user with their username and password. If the credentials are valid, a token is generated and returned to the user for authenticated sessions.

**Request Headers**

* Accept : application/json

**Request Body**

{

"username": "string",

"password": "string"

}

**3. Endpoint: User Logout**

* **URL**: /logout
* **Method**: GET
* **Description**: Logs out the authenticated user by revoking their current access token. This endpoint requires authentication using Sanctum.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.

**Request Body**

* This endpoint does not require a request body.

**4. Endpoint: Update Profile**

* **URL**: /user
* **Method**: POST
* **Description**: Updates the authenticated user's profile, including uploading a new profile picture. This endpoint requires authentication using Sanctum.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.
* Accept : application/json

**Request Body**  
  
profile\_picture (Optional): A file input containing the new profile picture to be uploaded.

**5. Endpoint: Get User Profile**

* **URL**: /user
* **Method**: GET
* **Description**: Retrieves the profile picture of the currently authenticated user. This endpoint requires authentication using Sanctum.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.

**Request Body**

This endpoint does not require a request body.

**. Endpoint: Get Banners**

* **URL**: /banner
* **Method**: GET
* **Description**: Retrieves a list of all banners.

**Request Headers**

* No special headers are required.

**Request Body**

* This endpoint does not require a request body.

**6. Endpoint: Create Banner**

* **URL**: /banner/create
* **Method**: POST
* **Description**: Creates a new banner. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.
* Accept : application/json

**Request Body**

The request body should be multipart/form-data containing the following fields:

[

            'title' => ['required', 'string'],

            'description' => ['required'],

            'status' => ['required'],

            'image' => ['required', 'image', 'mimes:png,jpg,jpeg'],

];

**7. Endpoint: Update Banner**

* **URL**: /banner/{id}
* **Method**: POST
* **Description**: Updates an existing banner identified by the banner ID. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.
* Accept : application/json

**URL Parameters**

* id (Required): The ID of the banner to be updated.

**Request Body**

The request body should be a JSON object containing the fields to be updated. For example:

[

            'title' ,

            'description' ,

            'status' ,

            'image' ,

];

**8. Endpoint: Delete Banner**

* **URL**: /banner/{id}
* **Method**: DELETE
* **Description**: Deletes an existing banner identified by the banner ID. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.

**URL Parameters**

* id (Required): The ID of the banner to be deleted.

**9. Endpoint: Get Blogs**

* **URL**: /blog
* **Method**: GET
* **Description**: Retrieves a list of all blogs. This endpoint is public and does not require authentication.

**Request Headers**

* None required.

**10. Endpoint: Create Blog**

* **URL**: /blog/create
* **Method**: POST
* **Description**: Creates a new blog entry. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.
* Accept : application/json

**Request Body**

The request body should be a JSON object containing the details of the blog to be created. For example:

[

            'image' => ['required', 'image', 'mimes:png,jpg,jpeg'],

            'title' => ['required'],

            'description' => ['required'],

            'tags' => ['required'],

            author => ['required'],

];

**11. Endpoint: Update Blog**

* **URL**: /blog/{id}
* **Method**: POST
* **Description**: Updates an existing blog entry. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.
* Accept : application/json

**Request Parameters**

* **Path Parameter**:
  + id (integer): The unique identifier of the blog to be updated.

**Request Body**

The request body should be a JSON object containing the updated details of the blog. For example:

[

            'image',

            'title' ,

            'description' ,

            'tags' ,

‘author’

];

**12. Endpoint: Delete Blog**

* **URL**: /blog/{id}
* **Method**: DELETE
* **Description**: Deletes an existing blog entry. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.

**Request Parameters**

* **Path Parameter**:
  + id (integer): The unique identifier of the blog to be deleted.

13. **Endpoint: Get Portfolios**

* **URL**: /portfolio
* **Method**: GET
* **Description**: Retrieves a list of all portfolios.

**14. Endpoint: Create Portfolio**

* **URL**: /portfolio/create
* **Method**: POST
* **Description**: Creates a new portfolio entry. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.

**Request Body**

[

            'title' => ['required'],

            'description' => ['required'],

            'image' => ['image', 'required', 'mimes:png,jpg,jpeg']

‘author’ => ‘required

];

**15. Endpoint: Update Portfolio**

* **URL**: /portfolio/{id}
* **Method**: POST
* **Description**: Updates an existing portfolio entry. This endpoint is restricted to admin users only.

**Request Headers**

* Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.

**Request Parameters**

* **Path Parameter**:
  + id (integer): The unique identifier of the portfolio to be updated.

**Request Body**

**Optional:**

[

            'title' ,

            'description' ,

            'image' ,

‘author’

];

**16. Endpoint: Delete Portfolio**

* **URL: /portfolio/{id}**
* **Method: DELETE**
* **Description: Deletes an existing portfolio entry. This endpoint is restricted to admin users only.**

**Request Headers**

* **Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.**

**Request Parameters**

* **Path Parameter:**
  + **id (integer): The unique identifier of the portfolio to be deleted.**

**17. Endpoint: Update User**

* **URL: /user/{username}**
* **Method: POST**
* **Description: Updates an existing user. This endpoint is restricted to admin users only.**

**Request Headers**

* **Authorization: Bearer {token} (Required)  
  The token obtained during login should be sent in the Authorization header.**

**Request Parameters**

* **Path Parameter:**
  + **id (integer): The unique identifier of the user to be updated.**

**Request Body (optional):**

**[**

**'name'**

**'username'**

**'email'**

**'password'**

**‘birth\_date’,**

**'phone\_number'**

**'profile\_picture'**

**];**