

# PAWSREUNITE - Backend Dev Testing

## 1. General Route

The screenshot shows the Postman interface with a successful response to the `HomePage` endpoint. The response body is:

```
1 "data": "Welcome to PawsReunite"
```

The dev server is running at port 3001  
GET `localhost:3001/`

Connected to dev database PawsReunite  
List number of the models  
GET `localhost:3001/databaseHealth`

Invalid Route  
GET `localhost:3001/aaaaaa`

The screenshot shows the Postman interface with a successful response to the `databaseHealth` endpoint. The response body is:

```
1 "data": {  
2     "databaseName": "PawsReunite",  
3     "databaseState": 1,  
4     "databaseHost": "127.0.0.1",  
5     "databaseModels": [  
6         "Role",  
7         "User",  
8         "Post",  
9         "Comment",  
10        "Notification"  
11    ]  
12 }  
13 }  
14 }
```

The screenshot shows the Postman interface with an invalid route response. The response body is:

```
1 "error": "Invalid Route"
```

## 2. Role Route

The screenshot shows the Postman interface with the 'PawsReunite' collection selected. In the left sidebar, under the 'Role' section, there is an item labeled 'GET Get All Roles'. The main panel displays the 'Get All Roles' request details. The method is 'GET', the URL is 'localhost:3001/roles', and the response body is shown in JSON format:

```
2 "data": [3 {4 "id": "64c606e1f4a57a8c4834b7de",5 "name": "regular",6 "description": "A regular user can view, create and read data.",7 "They can edit and delete only their own data.",8 "v": 09 },10 {11 "id": "64c606e1f4a57a8c4834b7df",12 "name": "admin",13 "description": "An admin user has full access and permissions to",14 "do anything and everything within this API.",15 "v": 016 },17 {18 "id": "64c606e1f4a57a8c4834b7e0",19 "name": "banned",20 }]
```

Return All Roles  
GET localhost:3001/roles

The screenshot shows the Postman interface with the 'PawsReunite' collection selected. In the left sidebar, under the 'Role' section, there is an item labeled 'GET Get Users from specific role'. The main panel displays the 'Get Users from specific role' request details. The method is 'GET', the URL is 'localhost:3001/roles/regular', and the response body is shown in JSON format:

```
1 "data": [2 {3 "id": "64c606e2f4a57a8c4834b7e4",4 "username": "eddy",5 "email": "eddy@gmail.com",6 "password": "$2b$10$ckJNlUN6xP5J0cJ36549.jfCQSwzFDITM78FEU7ut0d1Ea05yXi",7 "roleID": "64c606e1f4a57a8c4834b7de",8 "notifications": [],9 "v": 010 },11 {12 "id": "64c606e1f4a57a8c4834b7e0",13 }]
```

Return All Users from specific role  
GET localhost:3001/roles/regular

Params could be "regular, admin and banned"

### 3. User Route - Signup

The screenshot shows the Postman interface with a successful user registration. The request URL is `localhost:3001/users/signup`. The response body is a JSON object containing the user ID and a JWT token.

```
1 "userId": "64c708ae210f0ddf80f37ebc",
2 "JWTtoken": "eyJhbGciOiIzIiInR5cC16IkpXVC39.
eyJkYXRhIjoiZmEzmi2yjA2MwYNTFLOGfNzMzZmV1ZTdlMmQ2MTM5MWU0NzYyYmY2Mm
R1MDU0Y2I3M2FmNzU0NWEyMDMxM2hhNmRjYTlV10D2K2Y2YtT15zTY2MGV10TEOyVzJzDE4
YWNlMDc2M2I2YtJjNmV1YwFmNGM3ZTQ0NDMwODFKM2I4MTg3Y2Q4YzdKG13Yj1MG5NT
RhNTcxNzU4ZDiYzVmMzU2Y2I3NGU0Th1MTdkOTRKOT15ZGMxYzkxMGZmIiwiawFOijom
NjkwNzY1NDg2LCJleHA0jE20TAhTE40DZ9.
FQgXd5vfpBhEda2_sqyhF2gzHIPUtkUoqaWWhfsxM"
```

Successfully Signup  
POST `localhost:3001/users/signup`

Return error message  
when username is missing

Return error message  
when password is not strong

Return error message  
when username or password is already  
exist in Database

The screenshot shows a 500 Internal Server Error for a missing username. The request URL is `localhost:3001/users/signup`. The response body indicates validation failed due to a required username.

```
1 "error": "User validation failed: username: Path 'username' is required."
```

The screenshot shows a 400 Bad Request error for a password that is too short. The request URL is `localhost:3001/users/signup`. The response body lists an error message about password length.

```
1 "errors": [
2   "Password must be at least 8 characters long, contain at least 1
3   lowercase letter and 1 uppercase letter"
4 ]
```

The screenshot shows a 400 Bad Request error for existing user credentials. The request URL is `localhost:3001/users/signup`. The response body lists errors for both username and email already existing.

```
1 "errors": [
2   "Username already exists",
3   "Email already exists"
4 ]
```

### 3. User Route - Login

The screenshot shows the Paw application interface. On the left, the sidebar lists collections, environments, and history. Under the 'User' collection, the 'POST Sign in' endpoint is selected. The main panel shows a POST request to 'localhost:3001/users/signin'. The body contains JSON data: { "email": "test1@gmail.com", "password": "Abcd1984" }. The response status is 200 OK, and the response body is a complex JSON object containing user ID and a JWT token.

Successfully Signin  
POST [localhost:3001/users/signin](http://localhost:3001/users/signin)

Return UserId and JWT

Return error message  
when username is correct, but password is wrong

The screenshot shows the Paw application interface. The 'User' collection is selected. The 'POST Sign in' endpoint is selected. The body contains JSON data: { "email": "aaa@gmail.com", "password": "123456" }. The response status is 404 Not Found, and the response body is a JSON object with the message "error: 'User does not exist'".

Return error message  
when can't find userame

The screenshot shows the Paw application interface. The 'User' collection is selected. The 'POST Sign in' endpoint is selected. The body contains JSON data: { "email": "ii@gmail.com", "password": "1256" }. The response status is 400 Bad Request, and the response body is a JSON object with the message "error: 'Password is incorrect'".

### 3. User Route - Update User Detail

The screenshot shows the Paw application interface. On the left, the sidebar lists collections, environments, and history. Under the 'User' collection, the 'PUT Update user detail' endpoint is selected. The main panel shows a PUT request to 'localhost:3001/users/'. The body contains the following JSON:

```
1 {"username": "test2",  
2 "email": "test2@gmail.com",  
3 "password": "Bbz2907854"}  
4  
5
```

The response status is 200 OK, with a duration of 114 ms and a size of 1.34 KB. The response body is:

```
1 {  
2   "message": "User details updated successfully",  
3   "updatedDetails": {  
4     "username": "test2",  
5     "email": "test2@gmail.com"  
6   },  
7   "newJWT": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
8       eyJkYXRhIjoiZmEzMjIYjA2MWWyNTERFjN2MzMvIzTdlMmQ2MTNmYmM3ZGE4YWZmMG  
9       UyODAyODExZjAxMTFkZGZLZTMxZGMYjg10dk5OTU10WY1ZmQyMWZlZDQyNTUXZdg2YTNj  
10      N2M0NjJM2Q0NDgxYm3NmU4NTZmYzNLZt1iZTFmZmY5ZGR1Mjg20DAwZWlZZjk3MmJmN2  
11      Y2MzgyMGJjNjhIZCIsImIhdCI6MTY5MDc2NjYzOSwiZhwiJoNjkwODUzMMDM5EQ.  
12      z0MTNaS1I-bFn8B7pusABMYp43S0sTBtjgoTwns8I1Q"
```

Successfully Update User  
PUT localhost:3001/users/

The screenshot shows the Paw application interface. The 'User' collection is selected. The 'PUT Update user detail' endpoint is selected. The body contains the following JSON:

```
1 {"username": "bbbbbbb",  
2 "email": "bbbbbbb@gmail.com"}  
3  
4
```

The response status is 400 Bad Request, with a duration of 55 ms and a size of 924 B. The response body is:

```
1 {  
2   "errors": [  
3     "jwt expired"  
4   ]  
5 }
```

Return error message  
when JWT is not correct or missing

Return error message  
when email is not a valid email pattern  
the password validation will be the same  
as the signup route

The screenshot shows the Paw application interface. The 'User' collection is selected. The 'PUT Update user detail' endpoint is selected. The body contains the following JSON:

```
1 {"username": "",  
2 "email": "mail.com"}  
3  
4
```

The response status is 400 Bad Request, with a duration of 15 ms and a size of 947 B. The response body is:

```
1 {  
2   "errors": [  
3     "Please enter a valid email address"  
4   ]  
5 }
```

### 3. User Route - Delete user account

The screenshot shows the Paw API client interface. On the left, the sidebar lists collections, environments, and history. In the main area, a request is being made to the endpoint `PawsReunite / User / delete user account`. The method is set to `DELETE`, and the URL is `localhost:3001/users/64c708ae210f0ddf80f37ebc`. The `Auth` tab is selected, showing a `Bearer Token` type with a token value. The response status is `200 OK` with a `message`: "User deleted successfully".

Successfully Delete User  
DELETE localhost:3001/users/64c708ae210f0ddf80f37ebc

Return error message  
when user's account is not Admin account  
Only Admin has authorization to delete users

The screenshot shows the Paw API client interface. A request is being made to the endpoint `PawsReunite / User / delete user account` with the method set to `DELETE` and the URL `localhost:3001/users/64ab83cd1ed15812b0fb80f`. The `Auth` tab is selected, showing a `Bearer Token` type with a token value. The response status is `404 Not Found` with a message: "error": "User not found".

Return error message (User not found)  
when userID in params is not exist

The screenshot shows the Paw API client interface. A request is being made to the endpoint `PawsReunite / User / delete user account` with the method set to `DELETE` and the URL `localhost:3001/users/64ab83cd1ed15812b0fb80f`. The `Auth` tab is selected, showing a `Bearer Token` type with a token value. The response status is `400 Bad Request` with an array of errors: ["jwt expired", "Unauthorized"].

### 3. User Route - Get all users account

The screenshot shows the Postman interface with a successful API call. The URL is `localhost:3001/users/all`. The response body is a JSON array containing two user objects:

```
1
2 [
3   {
4     "_id": "64c606e2f4a57a8c4834b7e4",
5     "username": "eddy",
6     "email": "eddy@gmail.com",
7     "password": "$2b$10$ckjNlUN6xP5J0cJ365429.jfQoSowzFDITM78FEU7utdIea0syX",
8     "roleId": "64c606e1f4a57a8c4834b7de",
9     "notifications": [],
10    "__v": 0
11  },
12  {
13    "_id": "64c606e2f4a57a8c4834b7e5",
14    "username": "ji",
15    "email": "ji@gmail.com",
16    "password": "$2b$10$W090cG.F4E6910WwCo50HeZGwefYLif/rz1J6iOPriJyX5WM716S"
17  }
]
```

Successfully get all user  
GET `localhost:3001/users/all`

The screenshot shows the Postman interface with an error response. The URL is `localhost:3001/users/all`. The response body is a JSON object with an "errors" key:

```
1
2 {
3   "errors": [
4     "jwt expired",
5     "Unauthorized"
6   ]
}
```

Return error message  
when user's account is not Admin account  
Only Admin has authorization to get all users

# 4. Post Route - Get posts / post

PawsReunite

New Import

PUT Upd GET Get GET Filter GE > + ... No Environment

Collections Environments History

PawsReunite

General Request Role Post

GET Get ALL POSTS based on los... GET Filter

DEL Delete post by post id

POST Create new post

PUT Update post (only text not i... GET Get all posts from single user

GET Get distinct breed based on ... User Comment Notification

GET PawsR... / P... / Get ALL POSTS based on lost or ...

Send

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

Body

Pretty Raw Preview Visualize JSON

200 OK 129 ms 6.9 KB Save as Example

```
data: [
  {
    "_id": "64c6e862210f0ddf80f37df",
    "title": "sdssddssd",
    "species": "Dog",
    "breed": "Pug",
    "color": "Black",
    "description": "dssdsdsds",
    "photos": [
      "https://pawsreunite.s3.amazonaws.com/a3d014cb-3c90-494b-a260-40532d633aba-Poodle3.jpg"
    ],
    "suburb": "MOSMAN 2088",
    "contactInfo": "0433512626",
    "status": "found",
    "userId": "64c606e2f4a57a8c4834b7e5",
    "createdAt": "2023-07-30T22:46:58.058Z",
    "__v": 0
]
```

Successfully Get all posts  
GET localhost:3001/posts

PawsReunite

New Import

PUT Upd GET Get GET Filter GE > + ... No Environment

Collections Environments History

PawsReunite

General Request Role Post

GET Get ALL POSTS based on los... GET Filter

DEL Delete post by post id

POST Create new post

PUT Update post (only text not i... GET Get all posts from single user

GET Get distinct breed based on ... User Comment Notification

GET PawsR... / P... / Get ALL POSTS based on lost or ...

Send

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

status lost

species Cat

Body

Pretty Raw Preview Visualize JSON

200 OK 13 ms 2.58 KB Save as Example

```
data: [
  {
    "_id": "64c606e2f4a57a8c4834b7ec",
    "title": "Missing Cat",
    "species": "Cat",
    "breed": "Bengal",
    "color": "Multi",
    "description": "Our beloved cat, with a striking Bengal tiger-like color, has gone missing, and we desperately need your immediate assistance in locating our precious feline friend. This situation is urgent, and we are reaching out to the community for help. Our missing cat is a Bengal tiger-colored beauty with mesmerizing patterns on her fur. She has captivating green eyes that will capture your heart in an instant. She means the world to us, and we are deeply worried about her well-being.",
    "photos": [
      "https://pawsreunite.s3.ap-southeast-2.amazonaws.com/Bengali."
    ]
  }
]
```

Use Query Params to filter the posts  
Return list of posts which are satisfied with those condition or empty array when conditions are not satisfied.  
GET localhost:3001/posts?status=lost&species=Cat

PawsReunite

New Import

PUT Upd GET Get GET Filter GE > + ... No Environment

Collections Environments History

PawsReunite

General Request Role Post

GET Get ALL POSTS based on los... GET Filter

DEL Delete post by post id

POST Create new post

PUT Update post (only text not i... GET Get all posts from single user

GET Get distinct breed based on ... User Comment Notification

GET PawsR... / P... / Get ALL POSTS based on lost or ...

Send

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

postId 64c6e862210f0ddf80f37df3

Body

Pretty Raw Preview Visualize JSON

```
data: [
  {
    "_id": "64c6e862210f0ddf80f37df",
    "title": "sdssddssd",
    "species": "Dog",
    "breed": "Pug",
    "color": "Black",
    "description": "dssdsdsds",
    "photos": [
      "https://pawsreunite.s3.amazonaws.com/a3d014cb-3c90-494b-a260-40532d633aba-Poodle3.jpg"
    ],
    "suburb": "MOSMAN 2088",
    "contactInfo": "0433512626",
    "status": "found",
    "userId": "64c606e2f4a57a8c4834b7e5",
    "createdAt": "2023-07-30T22:46:58.058Z",
    "__v": 0
]
```

Get one post by query params postId  
GET localhost:3001/posts?postId=64c06367dac0d785bb1e2f91

Return error message when postId is invalid

PawsReunite

New Import

PUT Upd GET Get GET Filter GE > + ... No Environment

Collections Environments History

PawsReunite

General Request Role Post

GET Get ALL POSTS based on los... GET Filter

DEL Delete post by post id

POST Create new post

PUT Update post (only text not i... GET Get all posts from single user

GET Get distinct breed based on ... User Comment Notification

GET PawsR... / P... / Get ALL POSTS based on lost or ...

Send

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

postId 64c6e862210f0ddf80f37df3

Body

Pretty Raw Preview Visualize JSON

```
error: "Post not found"
```

## 4. Post Route - Get all posts from user

The screenshot shows the Postman interface with a collection named "PawsReunite". A specific POST request titled "Get all posts from single user" is selected. The request URL is "localhost:3001/posts/user". The "Auth" tab is active, showing a "Bearer Token" type with a token value. The response status is 200 OK, with a response time of 35 ms and a size of 3.74 KB. The response body is displayed in JSON format, showing a single post entry for a dog named Max.

```
1  {
2     "data": [
3         {
4             "_id": "64c606e2f4a57a8c4834b7e8",
5             "title": "Urgent: Missing Dog",
6             "species": "Dog",
7             "breed": "Pug",
8             "color": "Black",
9             "description": "Our beloved Pug named Max went missing on July 10th, 2023, in the Elmwood Avenue area of Anytown. Max is a friendly and well-trained dog with a golden coat and a distinctive white patch on his chest. He is medium-sized and approximately 2 years old. Max was last seen wearing a blue collar with an identification tag.",
10            "photos": [
11                "https://pawsreunite.s3.ap-southeast-2.amazonaws.com/pug1.jpg",
12                "https://pawsreunite.s3.ap-southeast-2.amazonaws.com/pug2.jpg"
13            ]
14        }
15    ]
16}
```

Successfully Get all post from specific user from JWT  
GET [localhost:3001/posts/user](http://localhost:3001/posts/user)

The screenshot shows the Postman interface with the same collection and request setup as the first screenshot. However, the response status is now 400 Bad Request, with a response time of 33 ms and a size of 924 B. The response body is a JSON object containing an "errors" array with a single entry: "jwt expired".

```
1  {
2     "errors": [
3         "jwt expired"
4     ]
5 }
```

Return error message  
when user's JWT is expired or missing provide JWT

# 4. Post Route - Filter Options

The screenshot shows the PawReunite API documentation interface. On the left, there's a sidebar with sections for Collections, Environments, and History. The main area displays a 'Post / Filter' endpoint. The URL is set to 'localhost:3001/posts/filter?status=color'. The 'Params' tab is selected, showing a query parameter 'status' with the value 'color'. Below this, the 'Body' tab shows a JSON response with the key 'data' containing an array of strings: 'Black', 'Brown', 'Cream', 'Grey', 'Multi', and 'White'. The status code is 200 OK, and the response time is 1232 ms.

Get distinct values of the 'status' field from the filtered posts  
GET `localhost:3001/posts/filter?status=color`

The screenshot shows the PawReunite API documentation interface. The left sidebar is identical to the first screenshot. The main area displays a 'Get distinct breed based on species' endpoint. The URL is set to 'localhost:3001/posts/distinct-breeds'. The 'Body' tab shows a JSON response with three objects, each representing a species and its breeds. The first object is for 'Cat' (breeds: Ragdoll, Siamese, Bengal). The second object is for 'Rabbit' (breeds: Other). The third object is for 'Pug' (breeds: Pug).

Get distinct breed based on species  
The purpose of this route is to limit the breed options only based on specific species  
GET `localhost:3001/posts/distinct-breeds`

# 4. Post Route - Create post

The screenshot shows the Paw app interface. On the left, the sidebar lists collections and environments. Under the 'Post' collection, the 'POST Create new post' endpoint is selected. The 'Body' tab is active, showing a form-data payload with fields: photos (cat9.jpg), title (Missing), species (Cat), and breed (Persian). Below the payload, the response status is 200 OK with a 3.93s duration and 1.25 KB size. The response body is displayed in JSON format:

```
2 "data": {  
3     "title": "Missing",  
4     "species": "Cat",  
5     "breed": "Persian",  
6     "color": "Black",  
7     "description": "this is a new post",  
8     "photos": [  
9         "https://pawsreunite.s3.amazonaws.com/  
10            62e0e63a-44d0-4fcd-9655-475799e440c3-cat9.jpg"  
11     ],  
12     "suburb": "2034",  
13 }
```

Successfully create post  
POST localhost:3001/posts

Return error message  
when request data is invalid

Return error message  
when required data is missing

Return error message  
when JWT expired or Missing JWT

The screenshot shows the Paw app interface. The 'Body' tab is active, showing a form-data payload with fields: photos (cat9.jpg), title (Missing), species (Ca), breed (Persian), color (Bla), and description (this is a new post). The response status is 400 Bad Request with a 1067 ms duration and 1.09 KB size. The error message in the response body is:

```
1 "error": "Post validation failed: species: 'Ca' is not a valid enum value  
for path 'species'., color: 'Bla' is not a valid enum value for path  
'color'., status: 'los' is not a valid enum value for path 'status'.",  
2  
3 [
```

The screenshot shows the Paw app interface. The 'Body' tab is active, showing a form-data payload with fields: photos (cat9.jpg), title (afgadd), species (cat), breed (english cat), color (black), and description (this is a new post). The response status is 400 Bad Request with a 1309 ms duration and 1.16 KB size. The error message in the response body is:

```
1 "error": "Post validation failed: title: Path 'title' is required.,  
species: Path 'species' is required., breed: Path 'breed' is required.,  
color: Path 'color' is required., suburb: Path 'suburb' is required.,  
contactinfo: Path 'contactinfo' is required., status: Path 'status'  
is required.",  
2  
3 [
```

The screenshot shows the Paw app interface. The 'Body' tab is active, showing a form-data payload with fields: color (black), description (this is a new post), suburb (Text 2034), status (lost), contactinfo (0433564758), and photos (Select files). The response status is 400 Bad Request with a 953 ms duration and 924 B size. The error message in the response body is:

```
1 "errors": [  
2     "jwt expired"  
3 ]  
4 [
```

# 4. Post Route - Update post

The screenshot shows the Postman interface with the following details:

- Collection:** PawsReunite
- Request Type:** PUT
- URL:** `localhost:3001/posts/64c71a2043138b530457903d`
- Body (form-data):**
  - title:** Missing puppy
  - description:** New description
  - status:** found
  - photos:** Parrot3.jpg (selected)
  - oldphotos:** (unchecked)
- Response:**

```
Pretty Raw Preview Visualize JSON
2 "data": {
3   "_id": "64c71a2043138b530457903d",
4   "title": "Missing puppy",
5   "species": "Cat",
6   "breed": "Persian",
7   "color": "Black",
8   "description": "New description",
9   "photos": [
10     "https://pawsreunite.s3.amazonaws.com/
```

Successfully update post  
PUT `localhost:3001/posts/64c71a2043138b530457903d`

Return error message  
when postId is not valid

Return error message  
when you are not authorised to update this post  
(this post is belong to other user)

Return error message  
when JWT expired or Missing JWT

The screenshot shows the Postman interface with the following details:

- Request Type:** PUT
- URL:** `localhost:3001/posts/64c71a2043138b530457903d`
- Auth:** Bearer Token (eyJhbGciOiJUzI1NiIsInR5cCI6I...
- Response:**

```
Pretty Raw Preview Visualize JSON
1
2   "error": "Post not found"
3 
```

The screenshot shows the Postman interface with the following details:

- Request Type:** PUT
- URL:** `localhost:3001/posts/64c71a2043138b530457903d`
- Auth:** Bearer Token (eyJhbGciOiJUzI1NiIsInR5cCI6I...
- Response:**

```
Pretty Raw Preview Visualize JSON
1
2   "error": "You are not authorized to update this post."
3 
```

The screenshot shows the Postman interface with the following details:

- Request Type:** PUT
- URL:** `localhost:3001/posts/64c71a2043138b530457903d`
- Auth:** Inherit auth from ...
- Response:**

```
Pretty Raw Preview Visualize JSON
1
2   "errors": [
3     "Unauthorized"
4   ]
5 
```

# 4. Post Route - Delete post

PawsReunite

DELETE localhost:3001/posts/64c606e2f4a57a8c4834b7e8

Auth Headers (7)

Type: Bearer Token

Token: eyJhbGciOiJIUzI1NilsInR5cCI6I...

Body: 200 OK 65 ms 1.72 KB

Response Body:

```
1 "data": {  
2     "_id": "64c606e2f4a57a8c4834b7e8",  
3     "title": "Urgent: Missing Dog",  
4     "species": "Dog",  
5     "breed": "Pug",  
6     "color": "Black",  
7     "description": "Our beloved Pug named Max went missing on July 10th, 2023, in the Elmwood Avenue area of Anytown. Max is a friendly and well-trained dog with a golden coat and a distinctive white patch on his chest. He is medium-sized and approximately 2 years old. Max was last seen wearing a blue collar with an identification tag.",  
8     "photos": [  
9         "https://pawsreunite.s3.ap-southeast-2.amazonaws.com/pug1.jpg",  
10        "https://pawsreunite.s3.ap-southeast-2.amazonaws.com/pug2.jpg",  
11    ]  
}
```

Successfully delete post  
DELETE localhost:3001/  
posts/64c606e2f4a57a8c4834b7e8

Return error message  
when you are not authorised to update this post  
(this post is belong to other user)  
Admin account can delete post as well

PawsReunite

DELETE localhost:3001/posts/64c606e2f4a57a8c4834b745

Auth Headers (7)

Type: Bearer Token

Token: eyJhbGciOiJIUzI1NilsInR5cCI6I...

Body: 404 Not Found 833 ms 922 B

Response Body:

```
1 "error": "Post not found"
```

Return error message  
when postId is not valid

PawsReunite

DELETE localhost:3001/posts/64c606e2f4a57a8c4834b7e7

Auth Headers (7)

Type: Bearer Token

Token: eyJhbGciOiJIUzI1NilsInR5cCI6I...

Body: 400 Bad Request 13 ms 924 B

Response Body:

```
1 "errors": [  
2     "jwt expired"  
3 ]
```

# 5. Comment Route - Comments

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, Environments, and History. The main area displays a request for 'PawsReunite / Comment / Get all comments'. The 'Params' tab shows a query parameter 'postId' with the value '64c606e2f4a57a8c4834b7ef'. The 'Body' tab shows a response with a status of 200 OK, 37 ms, and 1.66 KB. The response body is a JSON object with a 'data' key containing an array of comment objects. One comment object is expanded to show its details, including user information like '\_id', 'username', and 'email'.

Get all comments from post  
GET localhost:3001/comments?postId=64c606e2f4a57a8c4834b7ef

The screenshot shows the Postman application interface. The request path is identical to the one on the left: 'PawsReunite / Comment / Get all comments'. However, the response body is now an empty array, indicated by the number '1' and the text '["data": []]'.

If no comments under that post, return an empty array  
GET localhost:3001/comments?postId=64c606e2f4a57a8c4834b7ef

# 5. Comment Route - Comment

The screenshot shows the Postman interface for a collection named "PawsReunite". A GET request is selected with the URL `localhost:3001/comments/64c606e2f4a57a8c4834b7f5`. The response status is 200 OK, and the response body is displayed in JSON format:

```
1
2   "data": {
3     "_id": "64c606e2f4a57a8c4834b7f5",
4     "content": "this dog",
5     "userId": {
6       "_id": "64c606e2f4a57a8c4834b7e5",
7       "username": "ji",
8       "email": "ji@gmail.com",
9       "password": "$2b$10$WQ90cG.F4E6910WwCo50HeZGvewfYLif/rz1l6iOPriVvx5NW765",
10      "roleId": "64c606e1f4a57a8c4834b7df",
11      "notifications": [],
12      "v": 0
13    },
14    "postId": "64c606e2f4a57a8c4834b7ef",
15    "createdAt": "2023-07-30T06:44:48.379Z",
16    "v": 0
}
```

Get one comment from commentId  
GET `localhost:3001/comments/64c606e2f4a57a8c4834b7f5`

The screenshot shows the Postman interface for the same collection. A GET request is selected with the URL `localhost:3001/comments/64c606e2f4a57a8c4834b733`. The response status is 404 Not Found, and the response body is:

```
1
2   "error": "Comment not found"
3
```

Return error message when commentID is not valid

# 5. Comment Route - Create Comment

The screenshot shows the Paw application interface. On the left, there's a sidebar with collections like 'Post', 'User', and 'Comment'. In the main area, a request is being made to 'PawsReunite / Comment / Create comment' via a POST method to 'localhost:3001/comments/64ab8b5820c7076963872160'. The body of the request contains JSON with a single key 'content': 'Create new Comment'. The response status is 200 OK, and the response body is a JSON object containing a 'data' key with a detailed comment object.

```
POST /comments/64ab8b5820c7076963872160
{
  "content": "Create new Comment"
}

{
  "data": {
    "_id": "64c71ef1f8b2de06cd881869",
    "content": "Create new Comment",
    "userId": {
      "_id": "64c606e2f4a57a8c4834b7ed",
      "username": "eddy",
      "email": "eddy@gmail.com",
      "password": "$2b$10$ckJNlUN6xP51ocJ365429.",
      "jfcQsowzF0TTM78FEU7ut0d1Ea05yXi",
      "roleId": "64c606e1f4a57a8c4834b7de",
      "notifications": [],
      "__v": 0
    },
    "postId": "64ab8b5820c7076963872160",
    "createdAt": "2023-07-21T02:39:45.721Z"
  }
}
```

Successfully create comment under specific post  
POST [localhost:3001/comments/64ab8b5820c7076963872160](http://localhost:3001/comments/64ab8b5820c7076963872160)

Return error message when JWT expired or  
JWT missing

This screenshot shows a POST request to 'PawsReunite / Comment / Create comment' with an empty 'content' field. The response is a 500 Internal Server Error with the message 'Comment validation failed: Path `content` is required.'

```
POST /comments/64ab8b5820c7076963872160
{
  "content": ""
}

{
  "error": "Comment validation failed: Path `content` is required."
}
```

Return error message  
when content is empty

This screenshot shows a POST request to 'PawsReunite / Comment / Create comment' without a JWT token. The response is a 400 Bad Request with the message 'jwt expired'.

```
POST /comments/64ab8b5820c7076963872160
{
  "content": "Create new Comment"
}

{
  "errors": [
    "jwt expired"
  ]
}
```

# 5. Comment Route - Delete Comment

The screenshot shows the Paw API client interface. On the left, the sidebar lists collections like PawsReunite, General Request, Role, Post, User, and Comment. Under the Comment collection, there are several endpoints: GET Get all comments, GET Get specific comment by id, POST Create comment, and DEL Delete comment by id. The main panel shows a DELETE request to `localhost:3001/comments/64c74a3ea7f421e3a0d58299`. The 'Auth' tab is selected, showing a 'Bearer Token' type with a token value. The response body is a JSON object with the comment details.

```
1 "data": {  
2     "_id": "64c74a3ea7f421e3a0d58299",  
3     "content": "new comment",  
4     "userId": "64c606e2f4a57a8c4834b7e5",  
5     "postId": "64abbb5820c7076963872160",  
6     "createdAt": "2023-07-31T05:44:30.405Z",  
7     "__v": 0  
8 }
```

Successfully delete comment

DELETE `localhost:3001/comments/64c74a3ea7f421e3a0d58299`

Return error message when this comment is not belongs to you, unauthorised to delete this comment

The screenshot shows the Paw API client interface. The URL is `PawsReunite / Comment / Delete comment by id`. The 'Auth' tab is selected, showing a 'Bearer Token' type with a token value. The response status is 404 Not Found, and the body contains the error message "Comment not found".

```
1 2 3 "error": "Comment not found"
```

Return error message when commentId is not valid

The screenshot shows the Paw API client interface. The URL is `PawsReunite / Comment / Delete comment by id`. The 'Auth' tab is selected, showing a 'Bearer Token' type with a token value. The response status is 400 Bad Request, and the body contains the error message "You are not authorized to delete this comment".

```
1 2 3 "error": "You are not authorized to delete this comment"
```

# 6. Notification Route - Get all notifications and create notification

Get all notifications from specific user by validate JWT  
GET [localhost:3001/notifications](http://localhost:3001/notifications)

App will compare new created post with other posts once it found the matching, system will automatically create new notification based on found userId  
POST [localhost:3001/notifications](http://localhost:3001/notifications)

Return error message when JWT is expired or missing

POST [localhost:3001/notifications](http://localhost:3001/notifications)