Be Truth Speaker

All API's must have:

```
/api/v1
```

Facebook Endpoints

1. Add Facebook Post

   ```
   /add-face-post
   ```

2. Get Facebook Post

   ```
   /get-face-posts
   ```

   Return an array which contain seperately both `Approved` and `Pending` post

3. Get Single Post

   ```
   /get-face-post/:id
   ```

   Pass `id` in parameter to get Single Post Record from database

4. Delete Single Record/Post

```
/delete-face-post/:id
```

Pass `id` in parameter to get Delete Post Record from database

5. Delete All Post

```
/delete-all-face-posts
```

Will delete all records from the Facebook Collection exists in database

Instagram Endpoints | Twitter Endpoints | Tiktok Endpoints

Working is same as `Facebook Endpoints`

```
/add-insta-post
/get-insta-posts
/get-insta-post/:id
/delete-insta-post/:id
/delete-all-insta-posts
```

```
# Tiktok Endpoints
/add-tiktok-post
/get-tiktok-posts
/get-tiktok-post/:id
/delete-tiktok-post/:id
/delete-all-tiktok-posts
```

```
# Twitter Endpoints
/add-twitter-post
/get-twitter-posts
/get-twitter-post/:id
```

```
/delete-twitter-post/:id
/delete-all-twitter-posts
```

User Authentication

User Endpoints

1. Create User

   ```
   /register
   ```

2. Login

   ```
   /login
   ```

3. Logout

   ```
   /logout
   ```

```
// USER SETTINGS & PROFILE
router.route("/admin/update-password").put(adminHeader, updateUs

router.route("/admin/all-users").get(adminHeader, getAllUser);


router
    .route("/admin/user/:id")
    .get(adminHeader, getSingleUserByAdmin);
```

```
router
    .route("/admin/delete-user/:id")
    .delete(adminHeader, deleteUserByAdmin);

router
    .route("/admin/update-user/:id")
    .put(adminHeader, updateUserRoleByAdmin);
```

adminHeader Function

This code defines an `adminHeader` middleware function that is exported. Let's break down what this middleware does:

1. It is an asynchronous function (`catchAsyncErrors`) that takes three parameters: `req` (request object), `res` (response object), and `next` (a function to pass control to the next middleware).
2. Inside the middleware function, it defines an array `allowedPaths` containing paths that are accessible to admin users without requiring authentication. These paths include endpoints for retrieving posts (`/get-tiktok-posts`, `/get-face-posts`, `/get-twitter-posts`, `/get-insta-posts`) and adding a face post (`/add-face-post`).
3. It extracts the JWT token from the request headers (`req.headers.authorization`). If the token exists, it decodes the token using the JWT secret (`process.env.JWT_SECRET`) to obtain user information.
4. If decoding the token is successful and the decoded user's role is "admin", it attaches the user information (id, email, and role) to the request object (`req.user`) and calls `next()` to pass control to the next middleware or route handler.
5. If there's an error during token decoding or the decoded user is not an admin, it catches the error or simply proceeds to the next middleware without modifying the request object.