**API Documentation for Recipe App Backend**

**Base URL**

**http://localhost:5000/api**

**Authentication**

**1. Register a new user**

**POST /users/register**

| **Body Parameter** | **Type** | **Description** | **Required** | **Example** |
| --- | --- | --- | --- | --- |
| username | string | Unique username | Yes | **"john\_doe"** |
| email | string | Valid email address | Yes | **"john@example.com"** |
| password | string | Password (min 6 chars) | Yes | **"mypassword"** |

**Response:**

json

{

"message": "User registered",

"user\_id": 123

}

**2. Login**

**POST /users/login**

| **Body Parameter** | **Type** | **Description** | **Required** | **Example** |
| --- | --- | --- | --- | --- |
| username | string | Existing username | Yes | **"john\_doe"** |
| password | string | User password | Yes | **"mypassword"** |

**Response:**

json

{

"token": "JWT\_TOKEN\_HERE"

}

**Note:** Use this token in the **Authorization** header for protected routes:

text

x-auth-token: JWT\_TOKEN\_HERE

**User Operations**

**3. Update user**

**PUT /users/:id**

| **Path Param** | **Type** | **Description** |
| --- | --- | --- |
| id | int | User ID to update |

**Body:**

json

{

"email": "newemail@example.com", // optional

"password": "newpassword" // optional

}

**Response:**

json

{ "message": "User updated" }

**4. Delete user**

**DELETE /users/:id**

| **Path Param** | **Type** | **Description** |
| --- | --- | --- |
| id | int | User ID to delete |

**Response:**

json

{ "message": "User deleted" }

**Recipe Management**

**5. Add a recipe**

**POST /recipes**

| **Body Parameter** | **Type** | **Required** | **Description** |
| --- | --- | --- | --- |
| title | string | Yes | Recipe title |
| instructions | string | Yes | Preparation instructions |
| image\_url | string (URL) | Optional | Link to recipe image |
| ingredients | array of objects | Yes | List of ingredients with IDs and quantities |

**Ingredient object sample:**

json

{

"ingredient\_id": 1,

"quantity": "200g"

}

**Response:**

json

{

"message": "Recipe created",

"recipe\_id": 123

}

**6. Search recipes**

**GET /recipes/search**

Query parameters:

* **ingredient\_id** (optional, int) — filter recipes containing this ingredient

**Example:**

text

GET /recipes/search?ingredient\_id=5

**Response:**

json

[

// array of recipe objects

]

**Favorites Management**

**7. Add favorite**

**POST /favorites/:recipeId**

| **Path Param** | **Type** | **Description** |
| --- | --- | --- |
| recipeId | int | ID of the recipe to favorite |

**Response:**

json

{ "message": "Recipe added to favorites" }

**8. Remove favorite**

**DELETE /favorites/:recipeId**

| **Path Param** | **Type** | **Description** |
| --- | --- | --- |
| recipeId | int | ID of the recipe to remove favorite |

**Response:**

json

{ "message": "Removed from favorites" }

**9. List favorites**

**GET /favorites**

**Response:**

json

[

// array of favorite recipe objects

]

**Admin Operations *(Protected with Admin Role)***

**10. List users**

**GET /admin/users** (supports **page** and **limit** query params)

**11. Delete user**

**DELETE /admin/users/:id**

**12. Ban user**

**POST /admin/users/:id/ban**

**13. Suspend user**

**POST /admin/users/:id/suspend**

* Body: **{ "duration": 7 }** (days)

**14. Promote User to Admin**

**POST /admin/promote/:id**

| **Path Param** | **Type** | **Description** |
| --- | --- | --- |
| **id** | **int** | **User ID to promote to admin** |

**Response:**

**json**

**{**

**"message": "User promoted to admin"**

**}**

**15. Get Logistics / Dashboard Stats**

**GET /admin/logistics**

**Response example:**

**json**

**{**

**"totalUsers": 100,**

**"totalRecipes": 250**

**}**

**Notes**

* **All admin routes require an admin JWT token (Authorization: Bearer <token>).**
* **These endpoints perform user management, moderation, and analytics.**