IT Asset Management System API Endpoints

-By Krishna Rathi

Base url :- <http://127.0.0.1:5000/asset-management>

1. /v1/login

* **Method name** :- POST.
* **Description** :- used for authenticating and get access token and refresh token.
* **Headers** :- None
* **Parameter** :- None
* **Request Body** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| username | string, unique | User’s name | “user12” | Yes |
| password | string | User’s password | “User@12” | yes |

**Example :-**

{

"username" : "user",

"password" : "User@123"

}

* **Response** :-
  + **Success** :- HTTP status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | list containing data in dictionary | [ {} ] | Yes |
| message | String | On success message | User logged in successfully. | yes |

List of data contains:-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| access\_token | String – jwt | JWT access token | “zsx….” | Yes |
| refresh\_token | String – jwt | JWT refresh token | “sxe….” | yes |

**Example :-**

{

“success” : true,

"data" : [

{

"access\_token" : " ",

"refresh\_token" : " ",

}

]

“message” : "User logged in successfully "

}

* + **Error** :-
  + Error status code :- 401

{

“success” : false,

"data" : [ ]

“message” : "Invalid Credentials"

}

* + - Possible common error codes : 400 , 422, 500.
    - Refer error table.

1. /v1/logout

* **Method name** :- POST.
* **Description** :- used for logout and revoking access & refresh token.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Success Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | array | Empty array | [] | yes |
| message | String | On success message | User logged out successfully. | yes |

**Example** :-

{

“success” : true,

"data" : [ ],

“message” : "User logged out successfully "

}

* + **Error** :-
    - Possible error codes : 400, 401, 500.
    - Refer error table

1. /v1/refresh

* **Method name** :- POST.
* **Description** :- used for creating new refresh token and new access token when the access token of user is expired and previously issued refresh and access token would be revoked so the user would not have access to the previous tokens.
* **Headers** :- Authorisation header <Bearer refresh\_token> (JWT refresh token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- HTTP status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | list containing data in dictionary | [ {} ] | Yes |
| message | String | On success message | User logged in successfully. | yes |

List of data contains:-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| access\_token | String – jwt | JWT access token | “zsx….” | Yes |
| refresh\_token | String – jwt | JWT refresh token | “sxe….” | yes |

**Example :-**

{

“success” : true,

"data" : [

{

"access\_token" : " ",

"refresh\_token" : " ",

}

]

“message” : "Access and refresh token issued successfully"

}

* + **Error** :-
    - Possible common error codes : 400,401, 500.
    - Refer error table.

1. /v1/user/profile

* **Method name**:- GET
* **Description** :- used for getting user details can be accessed by logged in user only
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | List of dictionaries | [ {} ] | Yes |
| message | String | On success message | User profile displayed successfully. | yes |

List of data contains:-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| user\_id | String | Id of user | “EMPswd3” | yes |
| username | string | Username of user logged in. | “user12” | yes |

**Example** :-

{

“success” : true,

"data" : [

{

“user\_id” : "EMPswd3",

“username” : "user12 "

}

],

"message" : “User profile fetched successfully”

}

* + **Error** :-
    - Possible error codes : 401, 500.
    - Refer error table

1. /v1/user/change-password

* **Method name**:- PUT
* **Description** :- used by authenticated user to change password and default password given by admin at first time of login
* **Headers** :- Authorisation header <Bearer access\_token>
* **Parameter** :- None
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| new\_password | string | New password | “User@123” | Yes |
| confirm\_password | string | Confirm new password | “User@123” | yes |

**Example :-**

{

"new\_password" : "User@123",

"confirm\_password" : "User@123"

}

* **Response** :-
  + **Success** :- status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | array | Empty array | [] | yes |
| message | String | On success message | “Password changed successfully.” | yes |

**Example** :-

{

“success” : true,

"data" : [ ],

“message” : "Password changed successfully"

}

* + **Error** :-
  + Error status code 401

{

“success” : false,

"data" : [ ],

“message” : "Invalid credentials"

}

* + - Possible common error codes : 400, 422, 500.
    - Refer error table

1. /v1/users

* **Method name**:- GET
* **Description** :- used for getting all users in database. Accessed by admin and manager.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | array | List of dictionaries | [{}] | yes |
| message | String | On success message | User Fetched successfully. | yes |

List of data contains :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| user\_id | String | Id of user | “EMPswd3” | yes |
| username | string | Username of user to register | “user12” | yes |
| role | string | Role of user | “employee” /”asset manager “ | yes |

**Example** :-

{

“success” : true,

"data" : [

{

“user\_id” : "EMPswd3",

“username” : "user12 ",

“role” : "employee "

}

],

"message" : “Users fetched successfully”

}

* + **Error** :-
    - Possible error codes : 401, 403, 500.
    - Refer error table

1. /v1/users

* **Method name**:- POST
* **Description** :- used for creating new user in database. Accessed by admin only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| username | String, unique | User name | “user12” | Yes |
| role | string | Any role in system | “employee”/ “asset manager” | yes |

**Example :-**

{

"username" : "user12",

"role" : "employee"

}

* **Response** :-
  + **Success** :- status code :- 201 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | Empty list | [] | yes |
| message | String | On success message | “User created successfully.” | yes |

**Example** :-

{

“success” : true,

"data" : [ ],

"message" : “User created successfully”

}

* + **Error** :-
  + Error status code 409:-

{

“success” : false,

"data" : [ ],

"message" : “User already exists”

}

* + - Possible common error codes : 400, 401, 403, 422, 500.
    - Refer error table

1. /v1/categories

* **Method name**:- GET
* **Description** :- used for getting all categories in database. Accessed by admin and manager.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | List of dictionaries | [ {} ] | yes |
| message | String | On success message | “Categories fetched successfully.” | yes |

List of data contains :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| category\_id | String | Id of category | “CATswd3” | yes |
| category\_name | string | Name of category | “mouse” | yes |
| brand\_name | string | Brand name of categrory | “dell” | yes |
| vendor\_name | string | Vendor name of category | “Krishna” | yes |
| vendor\_email | string | Vendor email of category | “krish@gmail.com” | Yes |

**Example** :-

{

“success” : true,

"data" : [

{

“category\_id” : "CATswd3",

“category\_name” : "mouse ",

“brand\_name” : "dell",

“vendor\_name” : "Krishna",

“vendor\_email” : "krish@gmail.com"

}

],

"message" : “Categories fetched successfully”

}

* + **Error** :-
    - Possible error codes : 401, 403, 500.
    - Refer error table

1. /v1/categories

* **Method name**:- POST
* **Description** :- used for creating new category in database. Accessed by admin and manager only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| category\_name | string | Name of category | “mouse” | Yes |
| brand\_name | string | Brand name of category | “dell” | yes |
| vendor\_email | string | Email of vendor from which category purchased | “krish@gmail.com” | yes |

**Example :-**

{

"category\_name" : "mouse",

"brand\_name" : "dell",

"vendor\_email" : "krish@gmail.com"

}

* **Response** :-
  + **Success** :- status code :- 201 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | List | Empty list | [ ] | yes |
| message | string | On success message | “Category created successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ]

"message" : “Category created successfully”

}

* + **Error** :-
  + Error status code 409 :-

{

“success” : false,

"data" : [ ]

"message" : “Category already exists”

}

* + - Possible common error codes : 400, 401, 403, 422, 500.
    - Refer error table

1. /v1/vendors

* **Method name**:- GET
* **Description** :- used for getting all vendors in database. Accessed by admin and manager.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | List | List of dictionaries | [ {} ] | yes |
| message | string | On success message | “Vendors fetched successfully” | yes |

List of dictionary contains:-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| vendor\_id | String | Id of vendor | “VENswd3” | yes |
| vendor\_name | string | Name of vendor | “Krishna” | yes |
| vendor\_email | string | Email of vendor from which category purchased | “krish@gmail.com” | Yes |
| active\_status | bool | Status of vendor active or deactive | True/False | yes |

**Example** :-

{

“success” : true,

"data" : [

{

“vendor\_id” : "VENswd3",

“vendor\_name” : "Krishna",

“vendor\_email” : "krish@gmail.com",

“active\_status” : True

}

],

"message" : “Vendors fetched successfully”

}

* + **Error** :-
    - Possible error codes : 401, 403, 500.
    - Refer error table

1. /v1/vendors

* **Method name**:- POST
* **Description** :- used for creating new vendors in database. Accessed by admin and manager only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| vendor\_name | string | Name of vendor | “Krishna” | Yes |
| vendor\_email | string | Email of vendor | “krish@gmail.com” | yes |

**Example :-**

{

"vendor\_name" : "Krishna",

"vendor\_email" : "krish@gmail.com"

}

* **Response** :-
  + **Success** :- status code :- 201 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | List | Empty List | [ ] | yes |
| message | string | On success message. | “Vendor created successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ]

"message" : “Vendor created successfully”

}

* + **Error** :-
  + Error status code – 409

{

“success” : false,

"data" : [ ]

"message" : “Vendor already exists”

}

* + - Possible common error codes : 400, 401, 403, 422, 500.
    - Refer error table

1. /v1/vendor/{vendor\_id}

* **Method name**:- DELETE
* **Description** :- used for deactivating existing vendors in database. Accessed by admin only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- Path parameter :- vendor\_id: string
* **Response** :-
  + **Success** :- status code :- 200 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | Empty list | [ ] | yes |
| message | string | On success message. | “Vendor deactivated successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ],

“message” : "Vendor deactivated successfully"

}

* + **Error** :-
  + Error status code :- 404

{

“success” : false,

"data" : [ ],

“message” : "Vendor not exists"

}

* + - Possible error codes : 401, 403, 500.
    - Refer error table

1. /v1/assets

* **Method name**:- GET
* **Description** :- used for getting all assets in inventory. Accessed by manager.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | List | List of dictionaries | [ { } ] | yes |
| message | string | On success message. | “Assets fetched successfully” | yes |

List of data contains :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| asset\_id | String | Id of asset | “ASNswd3” | yes |
| category\_name | string | Category name of asset | “mouse” | yes |
| vendor\_name | string | Vendor name of asset | “krish@gmail.com | Yes |
| asset\_type | String | Type of asset | Assignable/ non - assignable | Yes |
| assigned\_to | string | Assigned to any particular user. | “EMPde34” | yes |
| asset\_status | string | Status of asset in the inventory. | Available/ unavailable/ under-maintenance | yes |

**Example** :-

{

“success” : true,

"data" : [

{

“asset\_id” : "ASNswd3",

“category\_name” : "mouse",

“vendor\_email” : "krish@gmail.com",

“asset\_type” : “assignable”,

“assigned\_to” : “EMPde34”,

“asset\_status” : “unavailable”

}

],

"message" : “Assets fetched successfully”

}

* + **Error** :-
    - Possible error codes : 400, 401, 500.
    - Refer error table

1. /v1/assets

* **Method name**:- POST
* **Description** :- used for adding assets in inventory . Accessed by manager only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| category\_name | string | Name of category | “laptop” | Yes |
| brand\_name | string | Brand name of category | “dell” | yes |
| vendor\_email | string | Email of vendor | “krish@gmail.com” | yes |
| asset\_type | string | Type of asset assignable or not | assignable | yes |

**Example :-**

{

"category\_name" : "laptop",

“brand\_name” : "dell",

"vendor\_email" : "krish@gmail.com",

"asset\_type" : "assignable"

}

* **Response** :-
  + **Success** :- status code :- 201 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | [ ] | List of data | [ ] | yes |
| message | string | On success response message. | “Asset added to inventory successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ]

"message" : “Asset created successfully”

}

* + **Error** :-
    - Possible error codes : 400, 401, 403, 404, 422, 500.
    - Refer error table

1. /v1/assets/{asset\_id}/assign

* **Method name**:- PUT
* **Description** :- used for assigning assets in inventory to an user. Accessed by manager only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- Path Parameter :- asset\_id
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| asset\_id | string | Id of asset. | “ASN3d45” | Yes |
| employee\_id | string | Id of employee to assign asset. | “EMP434f” | yes |

**Example :-**

{

"asset\_id" : "ASN3d45",

"employee\_id" : "EMP434f"

}

* **Response** :-
  + **Success** :- status code :- 200 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | List of data | [ ] | yes |
| message | string | On success response message. | “Asset assigned successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ],

"message" : “Asset assigned successfully”

}

* + **Error** :-
  + Error status code 404:-

{

“success” : false,

"data" : [ ],

"message" : “Asset does not exists or User does not exists”

}

* + - Possible common error codes : 400, 401, 403, 422, 500.
    - Refer error table

1. /v1/assets/{asset\_id}/unassign

* **Method name**:- PUT
* **Description** :- used for unassigning assets in inventory from an user. Accessed by manager only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- Path Parameter :- asset\_id
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| asset\_id | string | Id of asset. | “ASN3d45” | Yes |
| employee\_id | string | Id of employee to assign asset. | “EMP434f” | yes |

**Example :-**

{

"asset\_id" : "ASN3d45",

"employee\_id" : "EMP434f"

}

* **Response** :-
  + **Success** :- status code :- 200 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | Empty list. | [ ] | yes |
| message | string | On success response message. | “Asset unassigned successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ]

"message" : “Asset unassigned successfully”

}

* + **Error** :-
  + Error status code :- 404

{

“success” : false,

"data" : [ ]

"message" : “Asset does not exist”

}

* + - Possible error codes : 400, 401, 403, 422, 500.
    - Refer error table

1. /v1/issues

* **Method name**:- GET
* **Description** :- used for getting all issues raised by user. Accessed by manager.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | Empty list. | [ ] | yes |
| message | string | On success response message. | “Issues fetched successfully” | yes |

List of Data contains :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| issue\_id | string | Id of issue | “ISNswd3” | yes |
| asset\_id | string | Id of asset. | “ASN5563” | yes |
| user\_id | string | Id of employee to assign asset. | “EMPde34” | yes |
| issue\_status | string | Status of issue either issue or resolved | “pending/resolved” | Yes |

**Example** :-

{

“success” : true,

"data" : [

{

“issue\_id” : "ISNswd3",

“asset\_id” : "ASN5563",

“user\_id” : "EMPde34",

“issue\_status” : “pending”

}

],

"message" : “Issues fetched successfully”

}

* + **Error** :-
    - Possible error codes : 400, 401, 404, 500.
    - Refer error table

1. /v1/issues

* **Method name**:- POST
* **Description** :- used for raising issues for maintenance. Accessed by authorised user only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- None
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| asset\_id | string | Id of asset for raising maintenance issue. | “ASN43dr” | Yes |

**Example :-**

{

"asset\_id" : "ASN43dr"

}

* **Response** :-
  + **Success** :- status code :- 201 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | List of data | [ ] | yes |
| message | string | On success response message. | “Issue raised successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ]

"message" : “Issue raised successfully”

}

* + **Error** :-
    - Possible error codes : 400, 401, 403, 422, 500.
    - Refer error table

1. /v1/issue/{user\_id}

* **Method name**:- GET
* **Description** :- used for getting all raised issues by user. Accessed by user that raised issues.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :- Path parameter :- user\_id
* **Response** :-
  + **Success** :- status code :- 200 OK
  + Success reponse body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | List of dictionary | [ {} ] | yes |
| message | string | On success response message. | “Issues fetched successfully” | yes |

List of data contains:-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| issue\_id | string | Id of issue | “ISN53dr” | yes |
| asset\_id | string | Id of asset. | “ASN43dr” | yes |
| issue\_status | string | Status of issue pending | “pending” | Yes |

**Example** :-

{

“success” : true,

"data" : [

{

"issue\_id" : "ISN53dr",

"asset\_id" : "ASN43dr",

"issue\_status" : "pending"

}

]

"message" : “Issue raised successfully”

}

* + **Error** :-
  + Error status code 404:-

{

“success” : false,

"data" : [ ]

"message" : “Issue for this user not exists”

}

* + - Possible common error codes : 400, 401, 500.
    - Refer error table

1. /v1/issue/{issue\_id}

* **Method name**:- PUT
* **Description** :- used for resolving raised issued. Accessed by manager only.
* **Headers** :- Authorisation header <Bearer access\_token> (JWT access token recieved at time of login)
* **Parameter** :-Path Parameter :- issue\_id
* **Request** :-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| issue\_id | string | Issue id. | “ISN43dr” | Yes |
| asset\_id | string | Asset id | “ASN4def” | yes |
| user\_id | string | user id | “EMP4def” | yes |
| issue\_status | string | resolved | “resolved” | yes |

**Example :-**

{

"issue\_id" : "ISN43dr",

"asset\_id" : "ASN43dr",

"user\_id" : "EMP43dr",

"issue\_status" : "resolved"

}

* **Response** :-
  + **Success** :- status code :- 200 OK
  + On success response body schema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| success | bool | Successful or unsuccessful response | True/False | yes |
| data | list | List of data. | [ ] | yes |
| message | string | On success response message. | “Issue resolved successfully” | yes |

**Example** :-

{

“success” : true,

"data" : [ ]

"message" : “Issue resolved successfully”

}

* + **Error** :-
    - Possible error codes : 400, 401, 403, 404, 422, 500.
    - Refer error table

**Errors :-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Description | Example | Mandatory |
| error\_code | int | Unsuccessful response error code. | 403 | Yes |
| status | string | Unsuccessful response error status. | Forbidden | yes |
| message | string | Description of error message. | “Access is restricted on this resource” | yes |

**Example Schema body:-**

{

"error\_code" : 403,

"status" : “Forbidden”,

“message” : "Access is restricted on this resource”,

}

**Possible error codes table:-**

|  |  |  |
| --- | --- | --- |
| Error Status code | Message | Description |
| 400 | Bad Request | Data format of request body is not supported |
| 401 | Unauthorised | Token Expired. |
| 403 | Forbidden | Access to this resource is restricted. |
| 404 | Resource not found. | Resource or data not found. |
| 409 | Resource Conflict | Resource with same data already exists. |
| 422 | Unprocessible Entity | Insufficient or wrong fields of data provided in request body. |
| 500 | Internal Server Error | Server not responding. |