**API DOCUMENTATION**

Role Base Access Control

Task Number: TMS-2

RBAC: Role-based access control (RBAC) is a method of restricting system access to authorized users based on their roles within an organization.

In Django, you can implement RBAC using several strategies, including Django's built-in permissions system and third-party packages.

Here’s an overview of how to implement RBAC in Django:

1.Login

[Request](#_kd6tjl5w5wjo)

[Response](#_8b7ijrpuamb8)

Aim: To create an api for Role Base Access Control that is also called as RBAC

Purpose: Giving what level of access to which role should have that means employee have some permissions and driver have some permissions and admin have all permissions

and vendor have some permissions and company have some permissions .

## 1. login

Authenticate the user with the system and obtain the auth\_token

1.First API end points: Here we can access the url in app urls.py that generates the tokens.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/api/token/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  username  password | string  string |

**api\_key**

api\_key must be sent with all client requests. The api\_key helps the server to validate the request source.

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "auth\_key": <refresh\_token>  “auth\_key” : <access\_token>  }  auth\_key (**string**) - all further API calls must have this key in header |
| 403 | {"error": "API key is missing."} |
| 400 | {"error": "Please provide username."} |
| 400 | {"error": "Please provide password."} |
| 401 | {"error": "Invalid API key."} |
| 401 | {"error": "Incorrect username or password."} |
| 500 | {"error": “internal server"} |

## 2.second API end point: Here we can access the second url in app urls.py that is used to refresh and generate the new access token.

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/api/token/refresh/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  refresh | string |

## Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  “auth\_key” : <access\_token>  }  auth\_key (**string**) - all further API calls must have this key in header |

3.Third API end point: Here we access all the employee permission urls in app urls.py to access all the employee permisssions

Employee permissions:

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST**  **GET** | api/api/employee/transport-request/  api/api/employee/call-driver/  api/api/employee/sos-panic-button/  api/api/employee/get-transport-status/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| Authorization | <Bearer Token> | String(Access token) |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {      "message": "POST transport request"  }  {      "message": "GET transport request"  }  } |
| 403 | {"error": "API key is missing."} |
| 400 | {"error": "Please provide username."} |
| 400 | {"error": "Please provide password."} |
| 401 | {"error":"detail": "Authentication credentials were not provided.} |
| 405 | {"error": "detail": "Method \"GET\" not allowed."} |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {      "message": "GET driver request"  }      {      "message": "POST driver request"  }  } |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {      "message": "POST SOS/panic request"  }    {      "message": "POST SOS/panic request"  }  } |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {       "message": "GET transport status"  }  {      "message": "POST transport status"  }  } |

4.Fourth API end point: Here we access all the Driver permission urls in app urls.py to access all the Driver permisssions

Driver permissions

## Request

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  username  password | string  string |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "auth\_key": <refresh\_token>  “auth\_key” : <access\_token>  }  auth\_key (**string**) - all further API calls must have this key in header |

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/api/token/refresh/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  refresh | string |

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST**  **GET** | api/api/driver/assigned-trips/  api/api/driver/call-employee/  api/api/driver/start-trip/  api/api/driver/end-trip/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| Authorization | <Bearer Token> | String(Access token) |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {       "message": "GET assigned trip"  }  {      "message": "POST assigned trip"  }  } |

5.Fifth API end point: Here we access all the admin permission urls in app urls.py to access all the admin permisssions

Admin permissions

## Request

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  username  password | string  string |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "auth\_key": <refresh\_token>  “auth\_key” : <access\_token>  }  auth\_key (**string**) - all further API calls must have this key in header |

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/api/token/refresh/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  refresh | string |

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/api/admin/live-tracking-vehicles/  api/api/admin/live-tracking-employees/  api/api/admin/management-of-routes-schedules-allocations/  api/api/admin/Vendor-allocation/  api/api/admin/Assignment-of-routes-and-contracts-to-vendor/  api/api/admin/vendor-performance/  api/api/admin/get-bills/  api/api/admin/generate-reports/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| Authorization | <Bearer Token> | String(Access token) |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {       "message": "Admin live tracking vehicles"  }  } |

6.sixth API end point: Here we access all the vendor permission urls in app urls.py to access all the vendor permisssions

Vendor permissions

## Request

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  username  password | string  string |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "auth\_key": <refresh\_token>  “auth\_key” : <access\_token>  }  auth\_key (**string**) - all further API calls must have this key in header |

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/api/token/refresh/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  refresh | string |

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET**  **POST** | api/api/vendor/live-tracking-vehicles/  api/api/vendor/get-bills/  api/api/vendor/generate-reports/ |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {       "message": " GET vendor live tracking vehicles"  }  {       "message": " POST vendor live tracking vehicles"  }  } |

7.seventh API end point: Here we access all the Company permission urls in app urls.py to access all the Company permisssions

Company Permissions

## Request

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  username  password | string  string |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "auth\_key": <refresh\_token>  “auth\_key” : <access\_token>  }  auth\_key (**string**) - all further API calls must have this key in header |

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/api/token/refresh/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **key** | **Values** |
| POST | Body(x-www-form-urlencoded)  refresh | string |

## Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/api/company/live-tracking-vehicles/  api/company/live-tracking-employees/  api/api/company/get-bills/  api/api/company/generate-reports/ |

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  {       "message": "company live tracking vehicles"  }  } |

Conclusion:

**Role Assignment**: Users are assigned to roles based on their job functions. Each role has specific permissions associated with it.

 **Role Permissions**: Permissions define what actions a user can perform on specific resources. Roles aggregate these permissions.

