

API User DOCS
Manual Guide

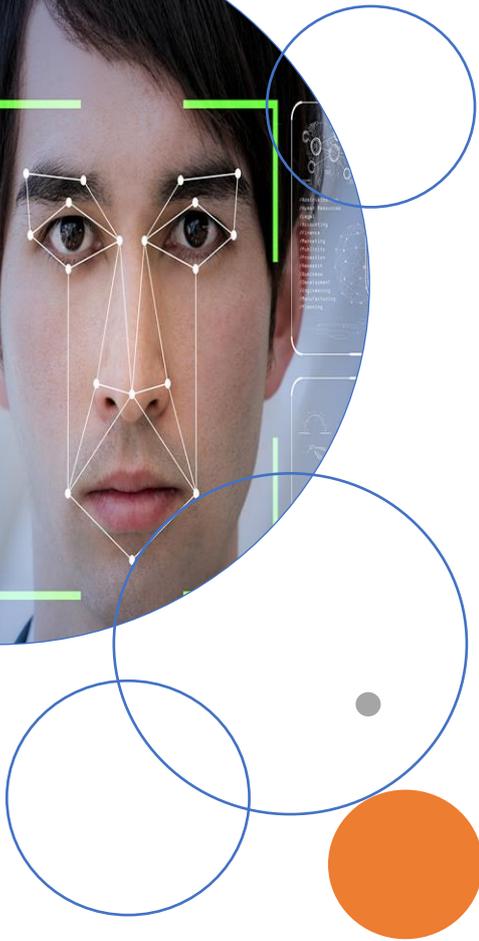
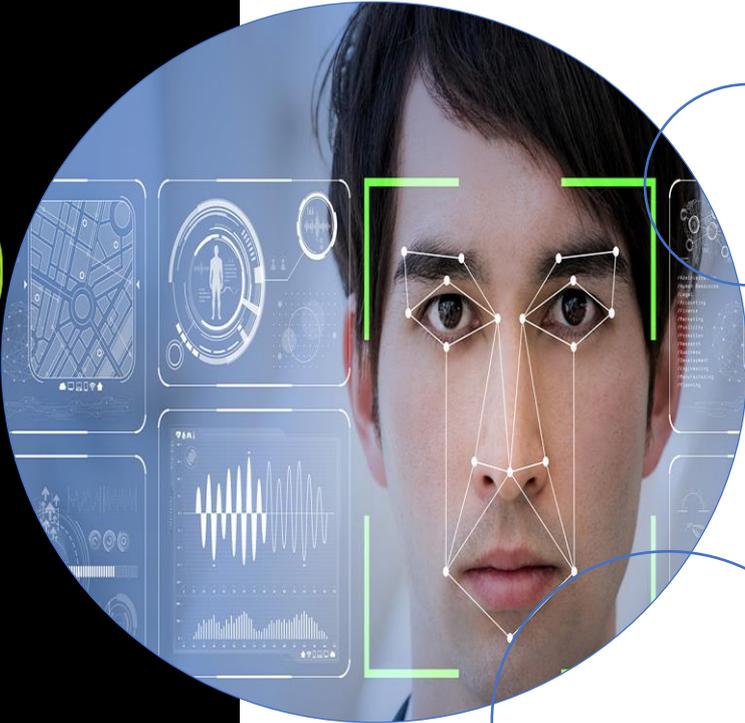


Table Contents:

Overview of BioMetrics	1
API DOCS.....	2
Personnel API DOCS.....	3
IClock API Docs.....	4

API DOCS MANUAL

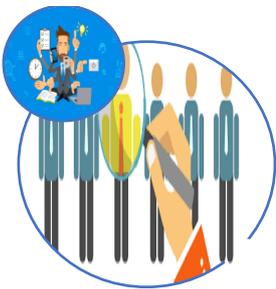
API Docs



This API will give Clear Information about the Authentication Token for Administrator and staff

URL: <http://127.0.0.1:8000/api/docs/>

Personnel API Docs



This API will give clear information about the Personnel Module Features Like

1. Employee
2. Departments
3. Areas
4. Positions
5. Locations

Iclock API Docs

This API will give clear information about the iclock Module Features Like

1. Terminals
2. Transactions

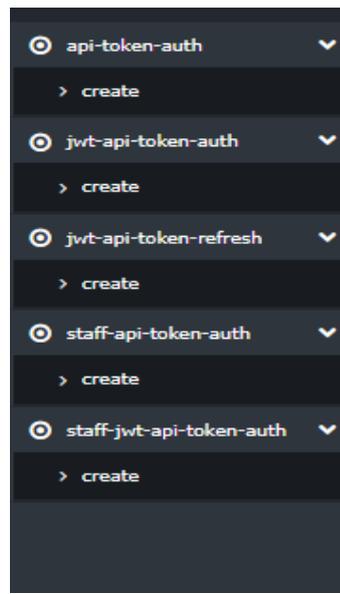
[http://127.0.0.1:8000 /api/iclock_docs/](http://127.0.0.1:8000/api/iclock_docs/)

1.API DOCS Authentication:

Our Products are configured with highly Secured and has rich in its Efficiency to Operate. When the term Secured is considered not only in terms of the Bio Metric Machine but also in software as well. Keeping this into consideration API's are designed in such a way that to Access the API's Requires Authentication Credentials

Customize in Almost No Time

URL for getting in to the Authentication Details are: <http://127.0.0.1:8000/api/docs/> then it shows the following features



DETAILED VIEW OF API DOCS Authentication

1.Why Token?

Usually an API token is a unique identifier of an application requesting access to your service. Your service would generate an API token for the application to use when requesting your service. You can then match the token they provide to the one you store to authenticate.

In Software Generally Consists of as follows

1.api-token-auth

2.jwt-api-token-auth

3.jwt-api-token-refresh

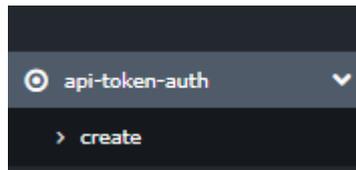
4.staff-api-token-auth

6.staff-jwt-api-token-auth

In all the above methods will using Create Method which is nothing but POST Method

1.api-token-auth

In this method will use create method



click on the create method to get

create

INTERACT

POST /api-token-auth/

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Click on Interact option

create

Username *

Awaiting request

Valid username for authentication

Password *

Valid password for authentication

CLOSE

SEND REQUEST

Enter the username and password click on the send request

Then Response will be as follows

POST

/api-token-auth/

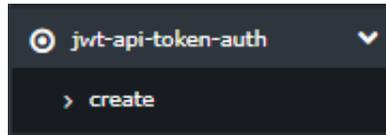
200

```
{
  "token": "f329a17c6689d4e9233b615b210e8ff132d97e1b"
}
```



This Token is Not Unique may vary for every login

2.jwt-api-token-auth:



click on the create method to get

create

INTERACT

POST /jwt-api-token-refresh/

API View that returns a refreshed token (with new expiration) based on existing token

If 'orig_jat' field (original issued-at-time) is found, will first check if it's within expiration window, then copy it to the new token

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Click on interact option

create

Username *

Awaiting request

Password *

CLOSE

SEND REQUEST

Enter the username and password click on the send request

DATA

RAW

POST /jwt-api-token-auth/

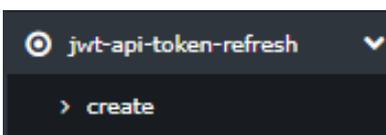
200

```
{
  .....
  "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2"
}
```

CLOSE

SEND REQUEST

3.jwt-api-token-refresh



click on the create method to get

jwt-api-token-refresh

create

INTERACT

POST /jwt-api-token-refresh/

API View that returns a refreshed token (with new expiration) based on existing token

If 'orig_iat' field (original issued-at-time) is found, will first check if it's within expiration window, then copy it to the new token

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Parameter	Description
token required	

Click on interact option

↔ create

Token *

Awaiting request

CLOSE

SEND REQUEST

Enter the username and password click on the send request

In token Enter the token which is generated in [jwt-api-token-auth](#) and click on send request so that will get a refreshed token

POST /jwt-api-token-refresh/ 200

```
{
  "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2Vybm91dCI6IjE2IiwiaWF0IjoiMTYxMjM0NTY3In0"
}
```

CLOSE

SEND REQUEST

4. staff-api-token-auth

staff-api-token-auth

> create

Click on the create method to get

staff-api-token-auth

create

INTERACT

POST /staff-api-token-auth/

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Click on interact option

create

Username * Awaiting request

Valid username for authentication

Password *

Valid password for authentication

[CLOSE](#) [SEND REQUEST](#)

This is Token is for Staff login the above tokens which are generated for administrator
After sending the response

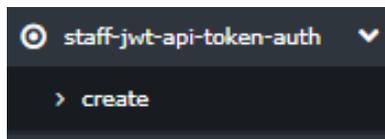
This is Token is for Staff login the above tokens which are generated for administrator
After sending the response

POST /staff-api-token-auth/ 200

```
{  
  "token": "f16cd9ffe9803d934fe2b440319b8a73442dcdd6"  
}
```

[CLOSE](#) [SEND REQUEST](#)

5. staff-jwt-api-token-auth



click on the create method to get

staff-jwt-api-token-auth

create [INTERACT](#)

POST /staff-jwt-api-token-auth/

API View that receives a POST with a user's username and password.
Returns a JSON Web Token that can be used for authenticated requests.

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Click on interact option

create

Username * Awaiting request

Valid username for authentication

Password *

Valid password for authentication

[CLOSE](#) [SEND REQUEST](#)

After Sending the response token will be generated

```
POST /staff-jwt-api-token-auth/ 200
{
  "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2"
}
```

Difference between token and Jwt token

JWT is an encoding standard for tokens that contains a JSON data payload that can be signed and encrypted.

JWT can be used for many things, among those are bearer tokens, i.e. a piece of information that you can present to some service that by virtue of you having it (you being the "bearer") grants you access to something.

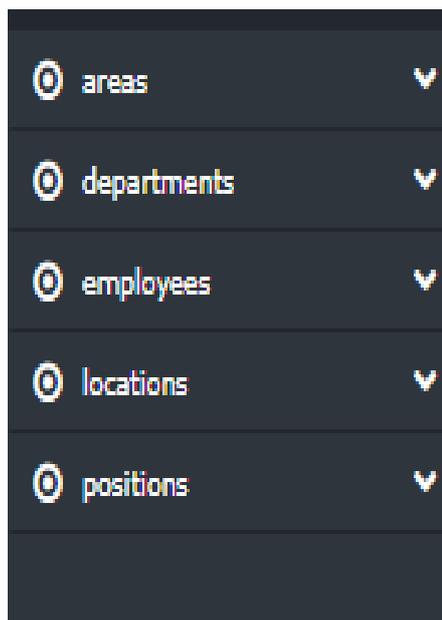
Bearer tokens can be included in an HTTP request in different ways, one of them (probably the preferred one) being the Authorization header. But you could also put it into a request parameter, a cookie, or the request body. That is mostly between you and the server you are trying to access.

Employees' related personnel information will be stored in personnel module. It consists of following API's

1. Areas
2. Departments
3. Positions
4. Locations
5. Employees

URL for getting into the Authentication Details are:

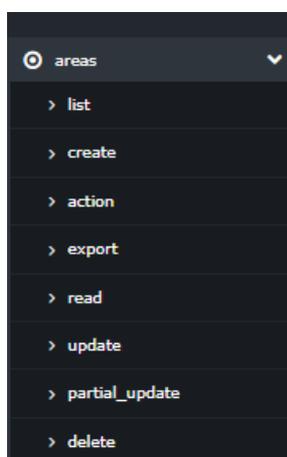
<http://127.0.0.1:8000/api/docs/>



Detailed View Personnel API:

1.Areas:

Areas Contains following Methods



a.List:

List is for to get all the areas present in the software

The URL to get all the areas from the software through the api is

list

GET /personnel/api/areas/

AreaList pagination, search, filter, order

Query Parameters

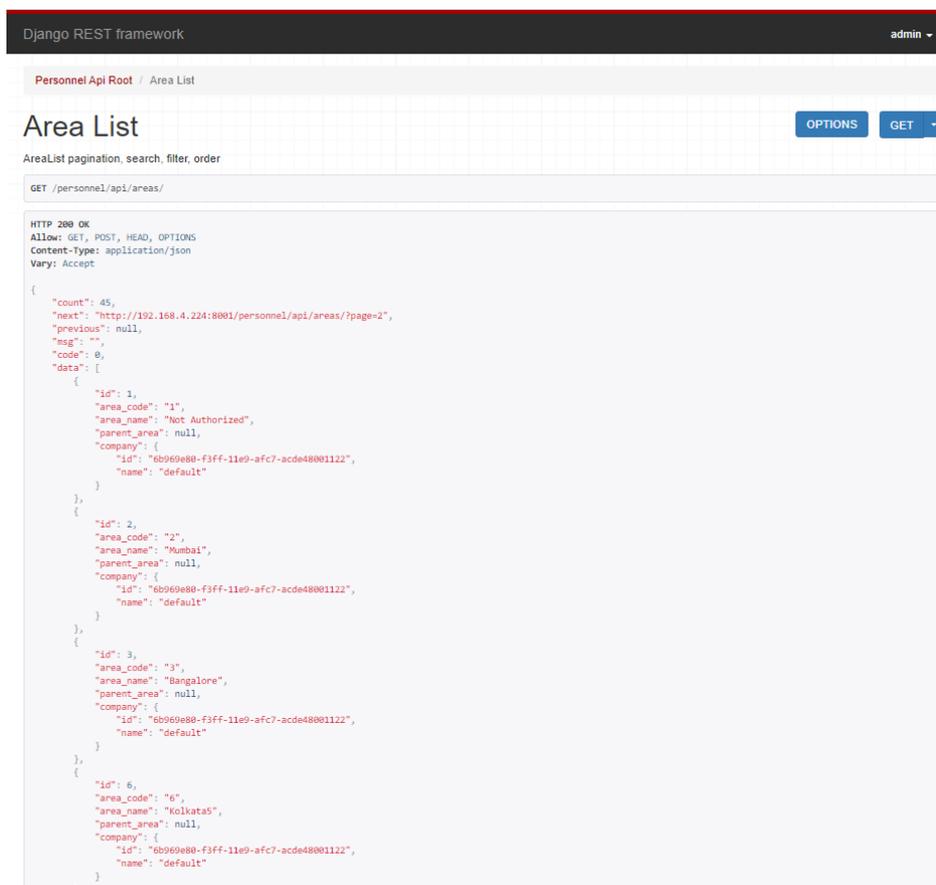
The following parameters should be included as part of a URL query string.

Base URL: 127.0.0.1:8001 End Url: /personnel/api/areas/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/personnel/api/areas/

Then it will display as follows:



The screenshot shows the Django REST framework API interface. At the top, it says "Django REST framework" and "admin". Below that, the breadcrumb is "Personnel Api Root / Area List". The main heading is "Area List" with "OPTIONS" and "GET" buttons. Below the heading, it says "AreaList pagination, search, filter, order" and "GET /personnel/api/areas/". The response is shown in a light blue box with the following JSON data:

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "count": 45,
  "next": "http://192.168.4.224:8001/personnel/api/areas/?page=2",
  "previous": null,
  "asg": "",
  "code": 0,
  "data": [
    {
      "id": 1,
      "area_code": "1",
      "area_name": "Not Authorized",
      "parent_area": null,
      "company": {
        "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
        "name": "default"
      }
    },
    {
      "id": 2,
      "area_code": "2",
      "area_name": "Mumbai",
      "parent_area": null,
      "company": {
        "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
        "name": "default"
      }
    },
    {
      "id": 3,
      "area_code": "3",
      "area_name": "Bangalore",
      "parent_area": null,
      "company": {
        "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
        "name": "default"
      }
    },
    {
      "id": 6,
      "area_code": "6",
      "area_name": "Kolkata5",
      "parent_area": null,
      "company": {
        "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
        "name": "default"
      }
    }
  ]
}
```

It displays the whichever areas present in the software

b.Create

Create Method is used to create the Area from Api this created area will reflect in software as well.

127.0.0.1:8001/personnel/api/areas/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

```
    }
  },
  {
    "id": 23,
    "area_code": "11",
    "area_name": "Andhra",
    "parent_area": {
      "id": 21,
      "area_code": "9",
      "area_name": "Mysore",
      "parent_area": null
    },
    "company": {
      "id": "6b969e89-f3ff-11e9-afc7-acde48801122",
      "name": "default"
    }
  }
]
}
```

Raw data HTML form

Area Code

Area Name

Parent

POST

Enter the details click on POST button the area with given details will be created.

c.Actions

Actions Method is used for some Action Purpose example Delete, Transfer etc. based on Features Actions are developed

action

⇌ INTERACT

POST /personnel/api/areas/action/

str(object=) -> str(str(bytes_or_buffer[, encoding[, errors]]) -> str

Create a new string object from the given object. If encoding or errors is specified, then the object must expose a data buffer that will be decoded using the given encoding and error handler. Otherwise, returns the result of object.str() (if defined) or repr(object). encoding defaults to sys.getdefaultencoding(). errors defaults to 'strict'.

Request Body

Click on Interact button

⇌ action

Object ids *

Awaiting request

Action type

delete

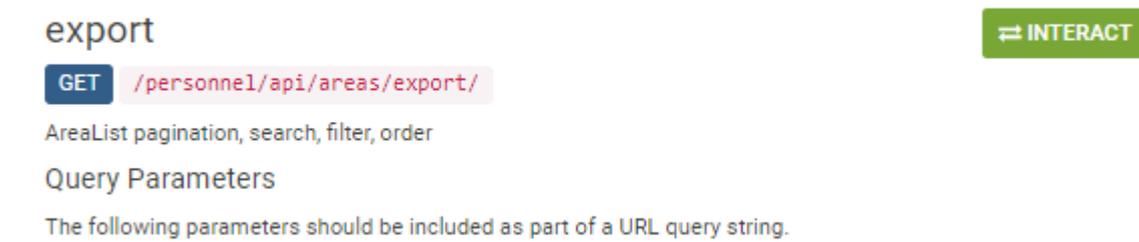
CLOSE SEND REQUEST

Enter the Area ID and click on Action Type and send Request.

d.Export

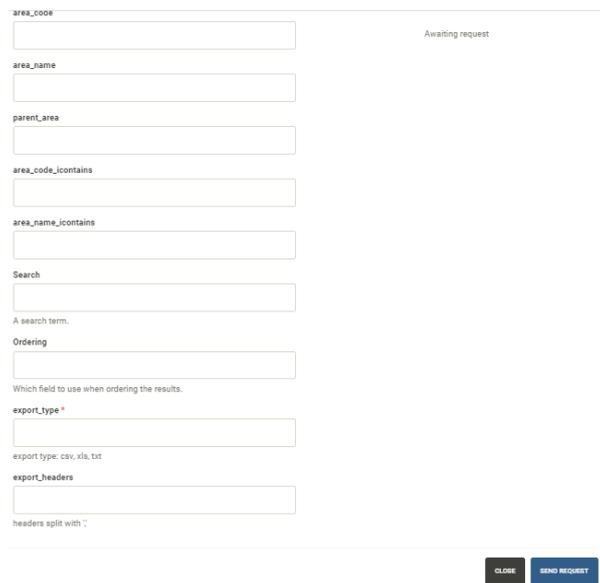
Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS



The screenshot shows an API documentation interface. At the top left, the word "export" is displayed in a large, dark font. To its right is a green button with a double-headed arrow and the text "INTERACT". Below "export", there is a blue button labeled "GET" followed by the URL "/personnel/api/areas/export/" in a light pink box. Underneath the URL, the text "AreaList pagination, search, filter, order" is shown. Below that, the heading "Query Parameters" is present, followed by the instruction "The following parameters should be included as part of a URL query string."

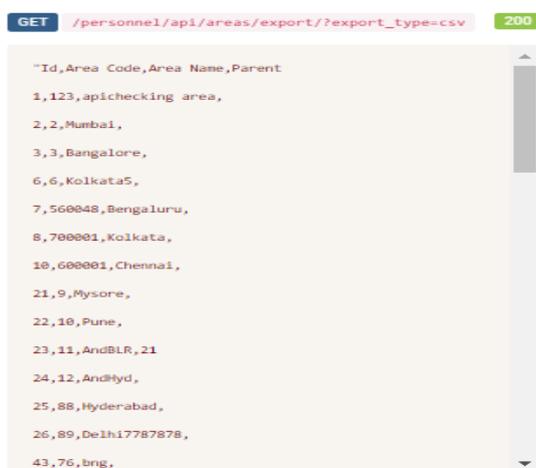
Enter the required fields and click on send request



The screenshot shows a form for sending an API request. The form is titled "Awaiting request" and contains several input fields: "area_code", "area_name", "parent_area", "area_code_contains", "area_name_contains", "Search", "Ordering", "export_type *", and "export_headers". Below the "export_type *" field, there is a note: "export type: csv, xls, txt". At the bottom of the form, there is a "headers split with:" field. At the bottom right of the form, there are two buttons: "CLOSE" and "SEND REQUEST".

After clicking on Send request data will be Exported.

Response will be



The screenshot shows the API response for the export endpoint. The response is a JSON array of objects, each representing an area. The response is displayed in a light yellow box with a scrollbar on the right. The response status is "200". The response data is as follows:

```
GET /personnel/api/areas/export/?export_type=csv 200
["Id,Area Code,Area Name,Parent
1,123,apichecking area,
2,2,Mumbai,
3,3,Bangalore,
6,6,Kolkata5,
7,560048,Bengaluru,
8,700001,Kolkata,
10,600001,Chennai,
21,9,Mysore,
22,10,Pune,
23,11,AndBLR,21
24,12,AndHyd,
25,88,Hyderabad,
26,89,De1h17787878,
43,76,brng,
```

e.Read

Read Is for to fetch the data based on the ID

read

INTERACT

GET /personnel/api/areas/{id}/

AreaList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Parameter	Description
id required	A unique integer value identifying this Area.

Click on the interact button and enter the id click on send request then the response will be as follows

read

DATA RAW

id *

A unique integer value identifying this Area.

GET /personnel/api/areas/1/

200

```
{
  "id": 1,
  "area_code": "123",
  "area_name": "apichecking area",
  "parent_area": null,
  "company": {
    "id": "6b969e80-f3ff-11e9-afc7-acde48801122",
    "name": "default"
  }
}
```

CLOSE

SEND REQUEST

f.Update

Update Method is Used for the update the details which is already created .for example if area1 is created if want to change the Full details of area1 then preferably choose update method.

update

INTERACT

PUT /personnel/api/areas/{id}/

AreaList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on interact button

Fill the details based on the id details which need to update and click on the send request then the request and response will be as follows.

↔ update DATA RAW

id * PUT /personnel/api/areas/2/ 200

A unique integer value identifying this Area.

Area Code

Area Name

Parent

```
{
  "id": 2,
  "area_code": "23",
  "area_name": "apichecking area",
  "parent_area": null
}
```

CLOSE SEND REQUEST

g. Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if area1 is created if want to change the particular details of area1 then preferably choose Partial update method.

partial_update ↔ INTERACT

PATCH /personnel/api/areas/{id}/

AreaList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact and enter the fields which ever need to update and click on send request.

↔ partial_update DATA RAW

id * PATCH /personnel/api/areas/2/ 200

A unique integer value identifying this Area.

Area Code

Area Name

Parent

```
{
  "id": 2,
  "area_code": "23",
  "area_name": "testingbfor api",
  "parent_area": null
}
```

CLOSE SEND REQUEST

The response will as mentioned above in the picture the fields which is changed is only one .if all fields need to be changed then preferably choose update method else partial update method.

h.Delete

Delete Method is used to delete the area which is already created.

delete ⇌ INTERACT

DELETE /personnel/api/areas/{id}/

AreaList pagination, search, filter, order

Path Parameters

Click on the interact button

Enter the Area which needs to delete then the response will be as follows

⇌ delete DATA RAW

id * **DELETE** /personnel/api/areas/74/ 200

74

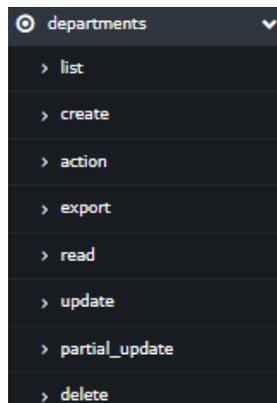
A unique integer value identifying this Area.

```
{
  ...
  "message": "deleted"
}
```

CLOSE SEND REQUEST

2.Departments:

Departments Contains following Methods



a.List:

List is for to get all the departments present in the software

The URL to get all the departments from the software through the api is

list ⇌ INTERACT

GET /personnel/api/departments/

DepartmentList pagination, search, filter, order

Query Parameters

Base URL: 127.0.0.1:8001 End Url: /personnel/api/departments/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/personnel/api/departments/

Click on interact button and click on send request to get all departments which are there in software

The screenshot shows a REST client interface with a 'list' endpoint. On the left, there are input fields for pagination (Page, Page size) and search (dept_code, dept_name, parent_dept, dept_code_icontains, dept_name_icontains, Search, Ordering). On the right, a GET request to '/personnel/api/departments/' is shown with a status of 200. The response is a JSON object with the following structure:

```
{
  "count": 20,
  "next": "http://192.168.4.224:8001/personnel/api/departments/?page=2",
  "previous": null,
  "msg": "",
  "code": 0,
  "data": [
    {
      "id": 1,
      "dept_code": "dfd",
      "dept_name": "cvvcvcv",
      "parent_dept": null,
      "parent_dept_name": null,
      "company": {
        "id": "6b969e88-f3ff-11e9-afc7-acde488",
        "name": "default"
      }
    },
    {
      "id": 2,
      "dept_code": "2",
      "dept_name": "testing",
      "parent_dept": null,
      "parent_dept_name": null,
      "company": {
        "id": "6b969e88-f3ff-11e9-afc7-acde488",
        "name": "default"
      }
    }
  ]
}
```

It displays the whichever departments present in the software

b.Create

Create Method is used to create the Department from Api. created department will reflect in software as well.

127.0.0.1:8001/personnel/api/departments/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

create

⇌ INTERACT

POST /personnel/api/departments/

DepartmentList pagination, search, filter, order

Request Body

Click on interact button fill the details which ever required to create the department and click on send request.

⇌ create

DATA RAW

Department Code *

456

Department Name *

apidepart

Parent

POST /personnel/api/departments/

201

```
{
  "id": 173,
  "dept_code": "456",
  "dept_name": "apidepart",
  "parent_dept": null
}
```

CLOSE

SEND REQUEST

Enter the details click on POST button the department with given details will be created.

c.Actions

Actions Method is used for some Action Purpose example Delete, Transfer etc. based on Features Actions are developed

Click on Interact button

Enter the Department ID and click on Action Type and send Request.

d.Export

Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS

export

⇌ INTERACT

GET /personnel/api/departments/export/

DepartmentList pagination, search, filter, order

Query Parameters

Enter the required fields and click on send request then the response will be as follows

export

GET /personnel/api/departments/export/?export_type=csv

1,dfg,cvccv,
2,2,testing,
3,3,Development,
4,4,tech support,
5,5,HR,
6,7,Dev,
9,6,Manager,
10,8,Test,
15,11,Cypress Department,
17,13,ElipseTesting,15
18,10,cytest2,
19,10,testing,
48,17924,AL,

export_type *
csv
export type: csv, xls, txt
export_headers
headers split with ;

CLOSE SEND REQUEST

e.Read

Read Is for to fetch the data based on the ID

read

INTERACT

GET /personnel/api/departments/{id}/

DepartmentList pagination, search, filter, order

Path Parameters

Click on the interact button and enter the id click on send request then the response will be as follows

read

DATA RAW

id *

5
A unique integer value identifying this Department.

GET /personnel/api/departments/5/ 200

```
{
  "id": 5,
  "dept_code": "5",
  "dept_name": "HR",
  "parent_dept": null,
  "parent_dept_name": null,
  "company": {
    "id": "6b969e80-f3ff-11e9-afc7-acde4801122",
    "name": "default"
  }
}
```

CLOSE SEND REQUEST

f.Update

Update Method is Used for the update the details which is already created. for example if department1 is created if want to change the Full details of department1 then preferably choose update method.

update

⇌ INTERACT

PUT /personnel/api/departments/{id}/

DepartmentList pagination, search, filter, order

Path Parameters

Click on interact button

Fill the details based on the id details which need to update and click on the send request then the request and response will be as follows.

⇌ update DATA RAW

id *

A unique integer value identifying this Department.

Department Code

Department Name

Parent

PUT /personnel/api/departments/163/ 200

```
{
  "id": 163,
  "dept_code": "789653",
  "dept_name": "namebydept",
  "parent_dept": null
}
```

CLOSE SEND REQUEST

g. Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if department1 is created if want to change the particular details of department1 then preferably choose Partial update method.

partial_update

⇌ INTERACT

PATCH /personnel/api/departments/{id}/

DepartmentList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact and enter the fields which ever need to update and click on send request.

⇌ partial_update DATA RAW

id *

A unique integer value identifying this Department.

Department Code

Department Name

Parent

PATCH /personnel/api/departments/166/ 200

```
{
  "id": 166,
  "dept_code": "1812",
  "dept_name": "api depart",
  "parent_dept": null
}
```

CLOSE SEND REQUEST

The response will as mentioned above in the picture the fields which is changed is only one .if all fields need to be changed then preferably choose update method else partial update method.

h.Delete

Delete Method is used to delete the department which is already created.

delete

⇌ INTERACT

DELETE /personnel/api/departments/{id}/

DepartmentList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path

Click on the interact button

Enter the Department which needs to delete then the response will be as follows

⇌ delete

DATA RAW

id *

DELETE /personnel/api/departments/174/

200

174

A unique integer value identifying this Department.

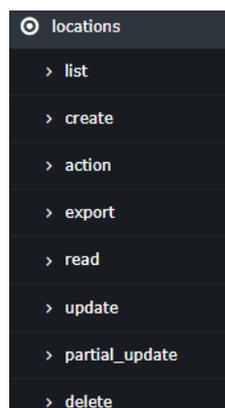
```
{
  "message": "deleted"
}
```

CLOSE

SEND REQUEST

3.Locations:

Locations Contains following Methods



a.List:

List is for to get all the locations present in the software

The URL to get all the locations from the software through the api is

list

INTERACT

GET /personnel/api/locations/

LocationList pagination, search, filter, order

Query Parameters

Base URL: 127.0.0.1:8001 End Url: /personnel/api/locations/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/personnel/api/locations/

Click on interact button and click on send request to get all locations which are there in software

The screenshot shows a REST client interface with a form on the left and a response preview on the right. The form includes fields for Page, Page size, location_code, location_name, parent_location, location_code_contains, location_name_contains, Search, and Ordering. The response preview shows a JSON object with a count of 17, a next URL, and a list of two location objects. The first location has id 1 and location code '1'. The second location has id 2 and location name 'Bangalore'.

```
{
  "count": 17,
  "next": "http://192.168.4.224:8001/personnel/api/lo...",
  "previous": null,
  "msg": "",
  "code": 0,
  "data": [
    {
      "id": 1,
      "location_code": "1",
      "location_name": "Location",
      "parent_location": null,
      "company": {
        "id": "8b960e88-f3ff-11e9-afc7-acd488",
        "name": "default"
      }
    },
    {
      "id": 2,
      "location_code": "2",
      "location_name": "Bangalore",
      "parent_location": null,
      "company": {
        "id": "8b960e88-f3ff-11e9-afc7-acd488",
        "name": "default"
      }
    }
  ]
}
```

It displays the whichever locations present in the software

b.Create

Create Method is used to create the location from Api. created location will reflect in software as well.

127.0.0.1:8001/personnel/api/locations/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

create

⇌ INTERACT

POST /personnel/api/locations/

LocationList pagination, search, filter, order

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Click on interact button fill the details which ever required to create the location and click on send request.

⇌ create DATA RAW

Location Code *

Location Name *

Parent

POST /personnel/api/locations/ 201

```
{
  "id": 59,
  "location_code": "123",
  "location_name": "akiveedu",
  "parent_location": null
}
```

CLOSE SEND REQUEST

Enter the details click on POST button the location with given details will be created.

c.Actions

Actions Method is used for some Action Purpose example Delete, Transfer etc. based on Features Actions are developed

Click on Interact button

Enter the Location ID and click on Action Type and send Request.

d.Export

Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS

export

⇌ INTERACT

GET /personnel/api/locations/export/

LocationList pagination, search, filter, order

Query Parameters

The following parameters should be included as part of a URL query string.

Enter the required fields and click on send request then the response will be as follows

export DATA RAW

location_code

location_name

parent_location

location_code_contains

location_name_contains

Search

A search term.

Ordering

Which field to use when ordering the results.

export_type *****

csv

export type: csv, xlsx, xls, txt

export_headers

headers split with ;

GET /personnel/api/locations/export/?export_type=csv 200

```
"Id,Location Code,Location Name,Parent
1,1,Location,
2,2,Bangalore,
3,3,Karnataka,
4,4,Kolkata,
5,5,Karnataka_Elpro,
6,6,WestBengal,
7,7,Telangana,
10,8,4,
11,9,xvxczv,1
12,10,retre,4
13,007,china,
19,001,SB5R,
21,123,Kolkata,6
55,9903,lucknow,
```

CLOSE SEND REQUEST

e.Read

Read Is for to fetch the data based on the ID

read

INTERACT

GET /personnel/api/locations/{id}/

LocationList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact button and enter the id click on send request then the response will be as follows

read DATA RAW

id *

2

A unique integer value identifying this Location.

GET /personnel/api/locations/2/ 200

```
{
  "id": 2,
  "location_code": "2",
  "location_name": "Bangalore",
  "parent_location": null,
  "company": {
    "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
    "name": "default"
  }
}
```

CLOSE SEND REQUEST

f.Update

Update Method is Used for the update the details which is already created. for example, if location1 is created if want to change the Full details of location1 then preferably choose update method.

update

INTERACT

PUT /personnel/api/locations/{id}/

LocationList pagination, search, filter, order

Path Parameters

Click on interact button

Fill the details based on the id details which need to update and click on the send request then the request and response will be as follows.

update DATA RAW

id *
A unique integer value identifying this Location.

Location Code

Location Name

Parent

PUT /personnel/api/locations/59/ 200

```
{
  "id": 59,
  "location_code": "123",
  "location_name": "Ajjamuru",
  "parent_location": null
}
```

CLOSE SEND REQUEST

g.Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if location1 is created if want to change the particular details of location1 then preferably choose Partial update method.

partial_update

INTERACT

PATCH /personnel/api/locations/{id}/

LocationList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URI path

Click on the interact and enter the fields which ever need to update and click on send request.

partial_update DATA RAW

id *
A unique integer value identifying this Location.

Location Code

Location Name

Parent

PATCH /personnel/api/locations/59/ 200

```
{
  "id": 59,
  "location_code": "123",
  "location_name": "Ajjamuru1",
  "parent_location": null
}
```

CLOSE SEND REQUEST

The response will as be mentioned above in the picture the fields which is changed is only one. if all fields need to be changed then preferably choose update method else partial update method.

h.Delete

Delete Method is used to delete the locations which is already created.

delete

⇌ INTERACT

DELETE /personnel/api/locations/{id}/

LocationList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Parameter	Description
id <small>required</small>	A unique integer value identifying this Location.

Click on the interact button

Enter the Location which needs to delete then the response will be as follows

↔ delete

DATA RAW

id *

DELETE /personnel/api/locations/59/

200

59

A unique integer value identifying this Location.

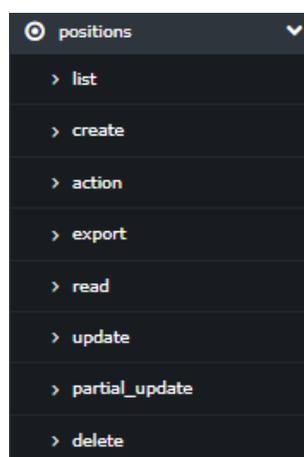
```
{
  "message": "deleted"
}
```

CLOSE

SEND REQUEST

4.Positions:

Positions Contains following Methods



a.List:

List is for to get all the Positions present in the software

The URL to get all the positions from the software through the api is

list ⇌ INTERACT

GET /personnel/api/positions/

PositionList pagination, search, filter, order

Base URL: 127.0.0.1:8001 End Url: /personnel/api/positions/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/personnel/api/positions/

Click on interact button and click on send request to get all positions which are there in software

The screenshot shows a REST client interface with a sidebar on the left containing various filters like Page, Page size, position_code, position_name, parent_position, position_code_contains, position_name_contains, Search, and Ordering. The main area displays a GET request to /personnel/api/positions/ with a 200 status. The response is a JSON object with a count of 37, a next URL, and a data array containing two position objects. The first object has id 1, position_code '1', position_name 'Position', and parent_position null. The second object has id 2, position_code '2', position_name 'Taster', and parent_position null. Both are associated with a company named 'default'.

It displays the whichever Positions present in the software

b.Create

Create Method is used to create the position from Api. created position will reflect in software as well.

127.0.0.1:8001/personnel/api/positions/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

create

⇌ INTERACT

POST /personnel/api/positions/

PositionList pagination, search, filter, order

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Click on interact button fill the details which ever required to create the position and click on send request.

⇌ create DATA RAW

Position Code *

Position Name *

Parent

POST /personnel/api/positions/ 201

```
{
  "position_code": "562",
  "position_name": "developmenttttttttttt",
  "parent_position": null
}
```

CLOSE SEND REQUEST

Enter the details click on POST button the Position with given details will be created.

c.Actions

Actions Method is used for some Action Purpose example Delete, Transfer etc. based on Features Actions are developed

Click on Interact button

Enter the Location ID and click on Action Type and send Request.

d.Export

Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS

export

⇌ INTERACT

GET /personnel/api/positions/export/

PositionList pagination, search, filter, order

Querv Parameters

Enter the required fields and click on send request then the response will be as follows

export

position_code

position_name

parent_position

position_code_contains

position_name_contains

Search

A search form.

Ordering

Which field to use when ordering the results.

export_type *

csr

export type: csr via: txt

export_headers

headers split with :

GET /personnel/api/positions/export/?export_type=csr

200

```

{"id": "Position Code,Position Name,Parent position,Emp",
  "1,1,Position,,0",
  "2,2,Tester,,2038",
  "3,3,Python Developer,,2033",
  "4,4,Java Developer,,1",
  "5,5,HR,,1",
  "6,6,Technical Support,,0",
  "7,7,HR and Admin,,2",
  "8,8,Manager,,3",
  "12,9,ETPprofessor,,1",
  "13,10,ETPDeveloper,,2",
  "14,11,ETPLead,,1",
  "15,12,ETPManager,,0",
  "16,13,ETPsupportingIneer,,1"}

```

CLOSE SEND REQUEST

e.Read

Read Is for to fetch the data based on the ID

read

INTERACT

GET /personnel/api/positions/{id}/

PositionList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact button and enter the id click on send request then the response will be as follows

read

DATA RAW

id *

102

A unique integer value identifying this Position.

GET /personnel/api/positions/102/

200

```

{
  "id": 102,
  "position_code": "562",
  "position_name": "developmenttttttttttt",
  "parent_position": null,
  "parent_position_name": null,
  "company": {
    "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
    "name": "default"
  }
}

```

CLOSE SEND REQUEST

f.Update

Update Method is Used for the update the details which is already created. for example, if position1 is created if want to change the Full details of position1 then preferably choose update method.

update

⇌ INTERACT

PUT /personnel/api/positions/{id}/

PositionList pagination, search, filter, order

Path Parameters

Click on interact button

Fill the details based on the id details which need to update and click on the send request then the request and response will be as follows.

update DATA RAW

PUT /personnel/api/positions/102/ 200

id *
102
A unique integer value identifying this Position.

Position Code *
562

Position Name *
clouddddddddddddddd

Parent

```
{  
  "position_code": "562",  
  "position_name": "clouddddddddddddddd",  
  "parent_position": null  
}
```

CLOSE SEND REQUEST

g. Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if position1 is created if want to change the particular details of position1 then preferably choose Partial update method.

partial_update

⇌ INTERACT

PATCH /personnel/api/positions/{id}/

PositionList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URI path

Click on the interact and enter the fields which ever need to update and click on send request.

partial_update DATA RAW

PATCH /personnel/api/positions/102/ 200

id *
102
A unique integer value identifying this Position.

Position Code *
cloud

Position Name

Parent

```
{  
  "position_code": "cloud",  
  "position_name": "cloud",  
  "parent_position": null  
}
```

CLOSE SEND REQUEST

The response will as be mentioned above in the picture the fields which is changed is only one. if all fields need to be changed then preferably choose update method else partial update method.

h. Delete

Delete Method is used to delete the positions which is already created.

delete

⇌ INTERACT

DELETE /personnel/api/positions/{id}/

PositionList pagination, search, filter, order

Click on the interact button

Enter the Position which needs to delete then the response will be as follows

↔ delete DATA RAW

id *

A unique integer value identifying this Position.

DELETE /personnel/api/positions/102/ 200

```
{
  "message": "deleted"
}
```

CLOSE SEND REQUEST

4. Employees:

Employees Contains following Methods

- employees
 - > list
 - > create
 - > action
 - > adjust_area
 - > adjust_depart
 - > adjust_emp_type
 - > adjust_position
 - > adjust_resign
 - > del_bio_template
 - > export
 - > import_emp
 - > read
 - > update
 - > partial_update
 - > delete

a.List:

List is for to get all the Employees present in the software

The URL to get all the Employees from the software through the api is

list

⇌ INTERACT

GET /personnel/api/employees/

EmployeeList pagination, search, filter, order

Base URL: 127.0.0.1:8001 End Url: /personnel/api/employees/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/personnel/api/employees/

Click on interact button and click on send request to get all employees which are there in software

The screenshot shows a REST client interface with a form on the left and a response preview on the right. The form includes fields for Page, Page size, Number of results to return per page, and various filters like emp_code, first_name, last_name, department, app_status, emp_code_contains, departments, status, first_name_contains, last_name_contains, area, and Search. There is also an Ordering field. The response preview shows a JSON object with a 'count' of 10000 and a 'data' array containing employee details.

```
{
  "count": 10000,
  "next": "http://127.0.0.1:8001/personnel/api/employees/?page=2",
  "previous": null,
  "page": 1,
  "order": 0,
  "data": [
    {
      "id": "85869716-cc8d-4569-803d-ba87",
      "emp_code": "M",
      "first_name": "Manager",
      "last_name": "Malya",
      "title_name": null,
      "doctor_password": null,
      "user_email": null,
      "department": {
        "id": 10,
        "dept_code": "M",
        "dept_name": "Mau"
      },
      "position": null,
      "time_data": null,
      "gender": null,
      "knowledge": null,
      "verify_code": 0
    }
  ]
}
```

It displays the whichever Employees present in the software

b.Create

Create Method is used to create the Employee from Api. created Employee will reflect in software as well.

127.0.0.1:8001/personnel/api/employees/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

create

⇌ INTERACT

POST /personnel/api/employees/

EmployeeList pagination, search, filter, order

Click on interact button fill the details which ever required to create the Employee and click on send request.

The screenshot shows a REST client interface with a form on the left and a response preview on the right. The form fields are: Employee ID * (7412333), First Name (vasudev), Last Name (billa), Local Name (prabhas), Device Password, Card No., Department * (1), Department ForeignKey, and Position. The response preview shows a JSON object with fields like id, emp_code, first_name, last_name, nickname, device_password, card_no, department, position, hire_date, gender, birthday, verify_node, emp_type, contact_tel, office_tel, mobile, national, city, address, postcode, email, enroll_no, ssn, religion, enable_att, enable_overtime, and enable_holiday.

Enter the details click on POST button the Position with given details will be created.

c.Actions

Actions Method is used for some Action Purpose example Delete, Transfer etc. based on Features Actions are developed

action

⇌ INTERACT

POST /personnel/api/employees/action/

str(object='') -> str str(bytes_or_buffer[, encoding[, errors]]) -> str

Click on Interact button

Enter the Location ID and click on Action Type and send Request.

d.Adjust_Area:

If Employee is transferred to Another area then Employee area will be adjusted by using adjust_area

adjust_area

⇌ INTERACT

POST /personnel/api/employees/adjust_area/

EmployeeList pagination, search, filter, order

Request Body

click on the adjust area and fill all the fields and click on send request then the request will be as follows.

⇌ adjust_area

Employees *

[ea7703a2-e2c9-11ea-a886-0242c0a87007]

Areas *

[122]

e.Adjust_Department:

If Employee is transferred to Another Department then Employee Department will be adjusted by using adjust_department

adjust_depart

⇌ INTERACT

POST /personnel/api/employees/adjust_depart/

EmployeeList pagination, search, filter, order

Request Body

The request body should be a "application/json" encoded object, containing the following items.

⇌ adjust_depart

Employees *

[0609]

Department *

1

f.Adjust_Position:

If Employee is transferred to Another Position then Employee Position will be adjusted by using adjust_position

adjust_position

⇌ INTERACT

POST /personnel/api/employees/adjust_position/

EmployeeList pagination, search, filter, order

g.Adjust_Resign:

If Employee is Resigning the Job. then Employee Resign will made through adjust_resign

adjust_resign

⇌ INTERACT

POST /personnel/api/employees/adjust_resign/

EmployeeList pagination, search, filter, order

Request Body

Fill the details and click on submit button

⇌ adjust_resign

Employees * Awaiting request

Resign date *

Resign type

Reason *

Disableatt *

h.Delete_bio_Template:

If Employee is bio-template needs to delete, then action will made through del_bio_template

del_bio_template

⇌ INTERACT

POST /personnel/api/employees/del_bio_template/

EmployeeList pagination, search, filter, order

Request Body

click on interact button and fill all the fields and submit the request then the response will be

⇌ del_bio_template

Employees * Awaiting request

Finger print

Face

Finger vein

Palm

i.Export & Import

Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS

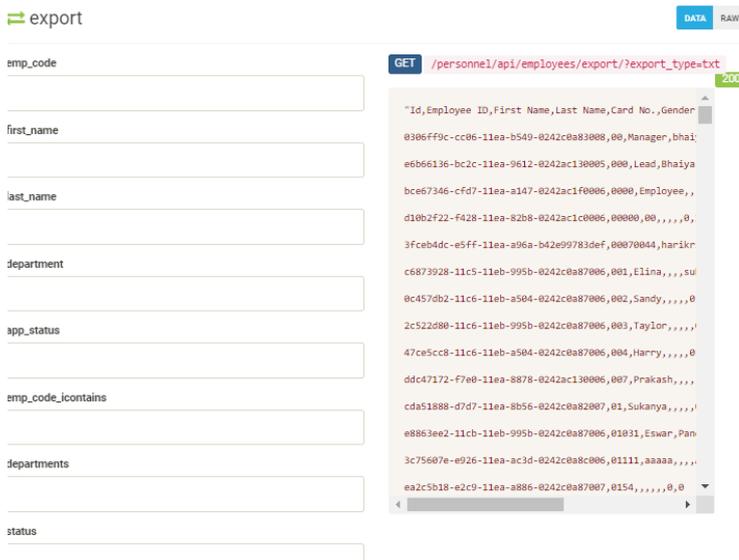
export

⇌ INTERACT

GET /personnel/api/employees/export/

EmployeeList pagination, search, filter, order

Enter the required fields and click on send request then the response will be as follows



The screenshot shows a REST client interface with a form on the left and a request/response pane on the right. The form has fields for: emp_code, first_name, last_name, department, app_status, emp_code_icontains, departments, and status. The request pane shows a GET request to `/personnel/api/employees/export/?export_type=txt`. The response pane shows a JSON array of 200 objects, each representing an employee with fields like Id, Employee ID, First Name, Last Name, Card No., Gender, and various identifiers.

Same way choose the file and import employee data

j.Read

Read Is for to fetch the data based on the ID

read

INTERACT

GET `/personnel/api/employees/{id}/`

EmployeeList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact button and enter the id click on send request then the response will be as follows

⇒ read

DATA RAW

id *

ea7703a2-e2c9-11ea-a886-0242c0a87007

A unique value identifying this Employee.

GET /personnel/api/employees/ea7703a2-e2c9-11ea-a886-0242c0a87007/ 200

```
{
  "id": "ea7703a2-e2c9-11ea-a886-0242c0a87007",
  "emp_code": "0685",
  "first_name": "",
  "last_name": null,
  "nickname": null,
  "device_password": null,
  "card_no": null,
  "department": {
    "id": 1,
    "dept_code": "dfd",
    "dept_name": "cvcvcv"
  },
  "position": null,
  "hire_date": null,
  "gender": null,
  "birthday": null,
  "verify_mode": 0,
  "emp_type": null,
  "contact_tel": null,
  "office_tel": null,
  "mobile": null,
  "national": null,
  "city": null,
  "address": null,
  "postcode": null,
  "email": null,
  "enroll_sn": "",
  "..."
}
```

CLOSE

SEND REQUEST

k.Update

Update Method is Used for the update the details which is already created. for example, if employee1 is created if want to change the Full details of employee1 then preferably choose update method.

update

PUT /personnel/api/employees/{id}/

EmployeeList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

⇒ INTERACT

Click on interact button

Fill the details based on the id details which need to update and click on the send request then the request and response will be as follows.

↔ update DATA RAW

id*

A unique value identifying this Employee.

Employee ID

First Name

Last Name

Local Name

Device Password

Card No.

Department

PUT /personnel/api/employees/ea7703a2-e2c9-11ea-a886-0242c0a87007/ 200

```
{
  "emp_code": "0605",
  "first_name": "dddddddddddddddd",
  "last_name": "bbbbbbbbbb",
  "nickname": "cccccccccccccccccccc",
  "device_password": null,
  "card_no": null,
  "department": 1,
  "position": null,
  "hire_date": null,
  "gender": null,
  "birthday": null,
  "verify_mode": 0,
  "emp_type": null,
  "contact_tel": null,
  "office_tel": null,
  "mobile": null,
  "national": null,
  "city": null,
  "address": null,
  "postcode": null,
  "email": null,
  "enroll_sn": "",
  "ssn": null,
  "religion": null,
  "enable_att": true,
  "enable_overtime": false,
  "enable_holiday": true,
  "dev_privilege": 0
}
```

I. Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if employee1 is created if want to change the particular details of employee1 then preferably choose Partial update method.

partial_update

↔ INTERACT

PATCH /personnel/api/employees/{id}/

EmployeeList pagination, search, filter, order

Path Parameters

Click on the interact and enter the fields which ever need to update and click on send request.

↔ partial_update DATA RAW

id*

A unique value identifying this Employee.

Employee ID

First Name

Last Name

Local Name

Device Password

Card No.

Department

PATCH /personnel/api/employees/ea7703a2-e2c9-11ea-a886-0242c0a87007/ 200

```
{
  "emp_code": "0605",
  "first_name": "eeeeeeeeeeee",
  "last_name": "bbbbbbbbbb",
  "nickname": "cccccccccccccccccccc",
  "device_password": null,
  "card_no": null,
  "department": 1,
  "position": null,
  "hire_date": null,
  "gender": null,
  "birthday": null,
  "verify_mode": 0,
  "emp_type": null,
  "contact_tel": null,
  "office_tel": null,
  "mobile": null,
  "national": null,
  "city": null,
  "address": null,
  "postcode": null,
  "email": null,
  "enroll_sn": "",
  "ssn": null,
  "religion": null,
  "enable_att": true,
  "enable_overtime": false,
  "enable_holiday": true,
  "dev_privilege": 0
}
```

The response will as be mentioned above in the picture the fields which is changed is only one. if all fields need to be changed then preferably choose update method else partial update method.

m.Delete

Delete Method is used to delete the employees which is already created.

delete

⇌ INTERACT

DELETE /personnel/api/employees/{id}/

EmployeeList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact button

Enter the Employee which needs to delete then the response will be as follows

⇌ delete

DATA RAW

id *

ea7703a2-e2c9-11ea-a886-0242c0a87007

A unique value identifying this Employee.

DELETE /personnel/api/employees/ea7703a2-e2c9-11ea-a886-0242c0a87007/ **200**

```
{
  "message": "deleted"
}
```

CLOSE

SEND REQUEST

2.ICLOCK API DOCS

Device' related information will be stored in iclock module. It consists of following API's

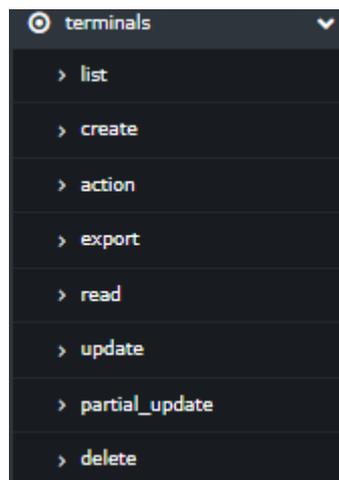
1.Terminals

2.Transactions

Detailed View Personnel API:

1.Terminals:

Terminals Contains following Methods:



a.List:

List is for to get all the devices present in the software

The URL to get all the devices from the software through the api is

list

GET /iclock/api/terminals/

TerminalList pagination, filter, order

⇌ INTERACT

Base URL: 127.0.0.1:8001 End Url: /iclock/api/terminals/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/iclock/api/terminals/

Then it will display as follows:

It displays the whichever areas present in the software

list DATA RAW

Page

A page number within the paginated result set.

Page size

Number of results to return per page.

sn

alias

ip_address

state

area

sn_contains

GET /iclock/api/terminals/ 200

```
{
  "count": 62,
  "next": "http://192.168.4.224:8001/iclock/api/terminal",
  "previous": null,
  "msg": "",
  "code": 0,
  "state": 1,
  "data": [
    {
      "id": 140,
      "sn": "8YEL192960005",
      "ip_address": "192.168.6.64",
      "alias": "Auto add",
      "terminal_name": null,
      "fw_ver": null,
      "push_ver": null,
      "state": 1,
      "terminal_tz": 330,
      "area": {
        "id": 1,
        "area_code": "123",
        "area_name": "apichecking area"
      },
      "last_activity": "2020-09-24 12:04:59",
      "user_count": null,
      "fp_count": null,
      "face_count": null,
      "palm_count": null,
      "transaction_count": null
    }
  ]
}
```

b.Create

Create Method is used to create the terminal from Api this created terminal will reflect in software as well.

127.0.0.1:8001/iclock/api/terminals/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

create INTERACT

POST /iclock/api/terminals/

TerminalList pagination, filter, order

Enter the details click on POST button the terminal with given details will be created.

create DATA RAW

Serial Number *

Device Name *

Device IP

Timezone

Connection Request Interval(sec)

Seconds

Transfer Mode

Transfer Interval

Minutes

POST /iclock/api/terminals/ 201

```
{
  "id": 167,
  "create_time": "2020-10-20 03:23:27",
  "create_user": null,
  "change_time": "2020-10-20 03:23:27",
  "change_user": null,
  "status": 0,
  "sn": "789654123",
  "alias": "qwerty",
  "ip_address": "192.168.1.12",
  "real_ip": null,
  "state": 1,
  "terminal_tz": 330,
  "heartbeat": 10,
  "transfer_mode": 1,
  "transfer_interval": 1,
  "transfer_time": "00:00:14:05",
  "product_type": 9,
  "is_attendance": 1,
  "is_registration": 0,
  "purpose": null,
  "controller_type": 0,
  "authentication": 1,
  "style": null,
  "upload_flag": "1111100000",
  "fw_ver": null,
  "push_protocol": "",
  "push_ver": null,
  "language": 84,
  "is_tft": false
}
```

c.Actions

Actions Method is used for some Action Purpose example Delete,Clear data etc. based on Features Actions are developed

action

INTERACT

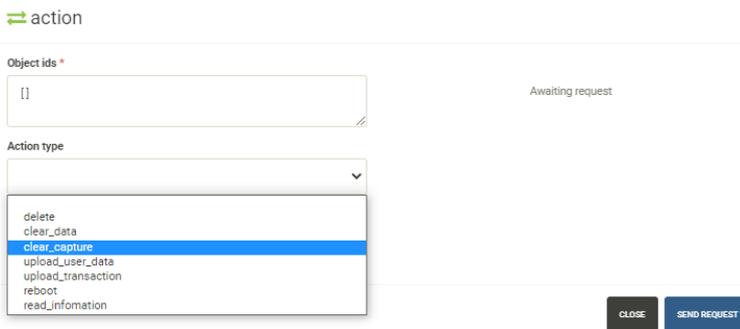
POST /iclock/api/terminals/action/

str(object=) -> str str(bytes_or_buffer[, encoding[, errors]]) -> str

Create a new string object from the given object. If encoding or errors is specified, then the object must expose a data buffer that will be decoded using the given encoding and error handler. Otherwise, returns the result of object.str() (if defined) or repr(object). encoding defaults to sys.getdefaultencoding(). errors defaults to 'strict'.

Request Body

Click on Interact button



Enter the object ID and click on Action Type and send Request.

d.Export

Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS

export

INTERACT

GET /iclock/api/terminals/export/

TerminalList pagination, filter, order

Query Parameters

Enter the required fields and click on send request



After clicking on Send request data will be Exported.

e.Read

Read Is for to fetch the data based on the ID

read

INTERACT

GET /iclock/api/terminals/{id}/

TerminalList pagination, filter, order

Path Parameters

Click on the interact button and enter the id click on send request then the response will be as follows

The screenshot shows a REST client interface for the endpoint `read`. The request is a GET to `/iclock/api/terminals/140/` with a status of 200. The response is a JSON object containing details for a terminal with ID 140.

```
{
  "id": 140,
  "sn": "BYEL192960005",
  "ip_address": "192.168.6.64",
  "alias": "Auto add",
  "terminal_name": null,
  "fw_ver": null,
  "push_ver": null,
  "state": 1,
  "terminal_tz": 330,
  "area": {
    "id": 1,
    "area_code": "123",
    "area_name": "apichecking area"
  },
  "last_activity": "2020-09-24 12:04:59",
  "user_count": null,
  "fp_count": null,
  "face_count": null,
  "palm_count": null,
  "transaction_count": null,
  "push_time": null,
  "transfer_time": "00:00;14:05",
  "transfer_interval": 1,
  "is_attendance": 1,
  "area_name": "apichecking area",
  "company": {
    "id": "6b969e80-f3ff-11e9-afc7-acde48001122",
    "name": "default"
  }
}
```

f.Update

Update Method is Used for the update the details which is already created .for example if terminal1 is created if want to change the Full details of terminal1 then preferably choose update method.

update

⇌ INTERACT

PUT /iclock/api/terminals/{id}/

TerminalList pagination, filter, order

Path Parameters

Click on interact button

Fill the details based on the id details which need to update and click on the send request .

g.Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if terminal1 is created if want to change the particular details of terminal1 then preferably choose Partial update method.

partial_update

⇌ INTERACT

PATCH /iclock/api/terminals/{id}/

TerminalList pagination, filter, order

Path Parameters

The following parameters should be included in the URL path.

Click on the interact and enter the fields which ever need to update and click on send request.

if all fields need to be changed then preferably choose update method else partial update method.

h.Delete

Delete Method is used to delete the area which is already created.

delete

⇌ INTERACT

DELETE /iclock/api/terminals/{id}/

TerminalList pagination, filter, order

Path Parameters

Click on the interact button

Enter the Area which needs to delete then the response will be as follows

⇌ delete DATA RAW

id * DELETE /personnel/api/areas/74/ 200

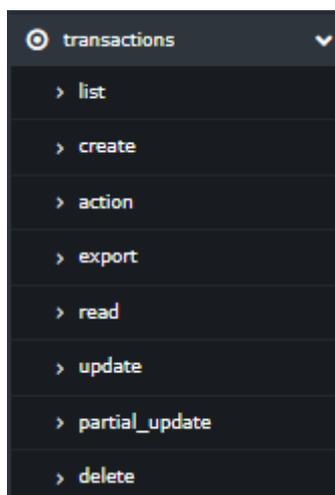
A unique integer value identifying this Area.

```
{
  "message": "deleted"
}
```

CLOSE SEND REQUEST

2.Transactions:

Transactions Contains following Methods



a.List:

List is for to get all the Transactions present in the software

The URL to get all the transactions from the software through the api is

list

GET /iclock/api/transactions/

TransactionList pagination, search, filter, order

Query Parameters

⇌ INTERACT

Base URL: 127.0.0.1:8001 End Url: /iclock/api/transactions/

In browser in need to consider Base URL + End URL i.e.,

127.0.0.1:8001/iclock/api/transactions/

Click on interact button and click on send request to get all departments which are there in software

Page

 A page number within the paginated result set.

Page size

 Number of results to return per page.

emp_code

terminal_sn

terminal_alias

start_time

end_time

GET /iclock/api/transactions/ 200

```
{
  "count": 23612,
  "next": "http://192.168.4.224:8001/iclock/api/transact
  "previous": null,
  "msg": "",
  "code": 0,
  "data": [
    {
      "id": 26665,
      "emp_code": "70047",
      "punch_time": "2020-01-30 09:14:06",
      "punch_state": "1",
      "verify_type": 1,
      "work_code": "0",
      "terminal_sn": "AEH2191360078",
      "terminal_alias": "Auto add",
      "area_alias": "Not Authorized",
      "longitude": null,
      "latitude": null,
      "gps_location": null,
      "mobile": null,
      "source": 4,
      "purpose": 9,
      "crc": "",
      "is_attendance": null,
      "reserved": null,
      "upload_time": "2020-09-09 12:25:43",
      "sync_status": null,
      "sync_time": null
    }
  ]
}
```

It displays the whichever Transactions present in the software

b.Create

Create Method is used to create the Transaction from Api. created Transaction will reflect in software as well.

127.0.0.1:8001/iclock/api/transactions/

Remember: For List and Create Method URL will same the only difference is for List will require GET Method and Create will require POST Method

create

⇌ INTERACT

POST /iclock/api/transactions/

TransactionList pagination, search, filter, order

Request Body

The request body should be a `Transaction` object containing the following items

Click on interact button fill the details which ever required to create the transaction and click on send request. Enter the details click on POST button the Transaction with given details will be created.

c.Actions

Actions Method is used for some Action Purpose example Delete.based on Features Actions are developed

Click on Interact button

Enter the Department ID and click on Action Type and send Request.

d.Export

Export is one of the key Feature in the software it will helps in Export All data present in the Particular Feature.

Data can be Export to Files CSV TXT XLS



Enter the required fields and click on send request then the response will be as follows



e.Read

Read Is for to fetch the data based on the ID



Click on the interact button and enter the id click on send request then the response will be as follows

read

DATA RAW

ID *

275

A unique integer value identifying this Transaction.

GET /iclock/api/transactions/275/ 200

```
{
  "id": 275,
  "emp_code": "A11",
  "punch_time": "2020-07-09 01:50:32",
  "punch_state": "255",
  "verify_type": 101,
  "work_code": "1",
  "terminal_sn": "App",
  "terminal_alias": null,
  "area_alias": null,
  "longitude": -122.084,
  "latitude": 37.4219983,
  "gps_location": "1600 Amphitheatre Pkwy, Mountain View, C",
  "mobile": "2",
  "source": 3,
  "purpose": 1,
  "crc": null,
  "is_attendance": 1,
  "reserved": null,
  "upload_time": "2020-07-09 14:28:36",
  "sync_status": 0,
  "sync_time": null,
  "temperature": null,
  "mask_flag": null,
  "company": "6b96e88-f3ff-11e9-afc7-acde48001122",
  "emp": "c25ea776-cca3-11ea-bf9c-0242c0a84006",
  "terminal": null
}
```

CLOSE SEND REQUEST

f.Update

Update Method is Used for the update the details which is already created. for example if transaction1 is created if want to change the Full details of transaction1 then preferably choose update method.

update

INTERACT

PUT /iclock/api/transactions/{id}/

TransactionList pagination, search, filter, order

Path Parameters

The following parameters should be included in the URI path

Click on interact button

Fill the details based on the id details which need to update and click on the send request.

g.Partial Update

Partial update Method is Used for the update the details which is required and those are already created .for example if transaction1 is created if want to change the particular details of transaction1 then preferably choose Partial update method.

partial_update

INTERACT

PATCH /iclock/api/transactions/{id}/

TransactionList pagination, search, filter, order

Path Parameters

Click on the interact and enter the fields which ever need to update and click on send request.If all fields need to be changed then preferably choose update method else partial update method.

h.Delete

Delete Method is used to delete the department which is already created.

delete

⇌ INTERACT

DELETE /iclock/api/transactions/{id}/

TransactionList pagination, search, filter, order

Path Parameters

Click on the interact button

Enter the Transaction which needs to delete then the response will be as follows

⇌ delete

ID *

275

A unique integer value identifying this Transaction.

