Employee Data

This collection contain APIS for adding employee and all employee data like employee office detail, bank detail, education, previous work experience, skills, dependents, projects and salary information. This module is created to add data of employee to maintain the record in the system.

Employee personal/office detail

This forlder contain APIS to perform CRUD on personal and office details of emoloyee.Personal detail is added using POST request and we need to pass personal in module serializer. Office detail is added using PATCH request and office is passed in module serializer. While adding office detail when add official email and press submit an hrms user id is created that will be unique for each employee.

Employee type router

This router is used to add Employee type. There are 2 employee types: Permanent and contract.

POST Post Employee Type

```
{{Url}}/api/employees/types/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title": "Permanent",
    "level": 1
}
```

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

```
{{Url}}/api/employees/types/
```

HEADERS

Authorization	{{bearerToken}}

PARAMS

GET Get specific Employee Type

HEADERS

Authorization	Bearer

PARAMS

POST Post Employee detail

```
{{Url}}/api/employees/06817ab0-b5f0-41df-8090-7e35f75f704b/
```

In post request personal detail data is added.

Cases:

1. While adding employee personal detail check by providing correct or incorrect org id in body.
2. While adding employee personal detail check by adding data in required and optional fields.
3. While adding employee personal detail check the impact by not adding data in required fields.
4. While adding employee personal detail check by not adding data in optional fields.
5. Check if after concatinating first name and last name proper name is appearing or not.
6. Check the format of dare field.
7. Cechk if there is a check of uniquness of CNIC no

7. Check if there is a check of uniquness of CNIC no.
8. Check by providing correct data in choise fields like gender, marital status and blood group.
9. Check all the formats in profile picture field.

Authorization	{{bearerToken}}

Body formdata

first_name	office
father_name	
last_name	detail
dob	2023-11-11
cnic_no	55555-5555556-9
blood_group	B+
passport_no	9876
date_of_expiry	2023-10-11
gender	2
marital_status	2
profile_image	
department	19
position	10
staff_classification	13
current_salary	123456
starting_salary	123455
employee_type	2
joining_date	2022-10-11
hiring_comment	as asd sda
leaving_date	2022-10-14
leaving_reason	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

hrmsuser	user1

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

skype	aassd12
status	active
organization	7
module	office
is_active	true
personal_email	asma5@hrms.com
official_email	test@mailiantor.com

GET List Employee Detail

```
{{Url}}/api/employees/
```

Cases:

1. Check if all the active data is appearing in get all API.

2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee.

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Authorization	{{bearerToken}}

GET Get specific Employee Detail

```
{{Url}}/api/employees/cf31ddca-00ac-4ac5-9202-96192749cc8c/
```

In URL pass Employee UUID.

Cases:

1. View a specific employee by providing inactive uuid.
2. View a specific employee by providing existing uuid.
3. View a specific employee by providing the uuid that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

DELETE Delete Employee Detail

```
{{Url}}/api/employees/11307219-3003-4dd4-86ab-497d44803fe0/
```

In URL pass Employee UUID.

Cases:

1. Check we are able to delete employee by providing it's id.
2. Check admin is able to delete employee of some other organization or not.
3. Check what happens if we delete deleted employee.

HEADERS

Authorization	{{bearerToken}}

PARAMS

PATCH Update Employee Detail

```
{{Url}}/api/employees/fcf82b91-cb0c-4d26-b342-1899c2e9e2e9/
```

In URL pass Employee UUID.

Cases:

1. While updating personal detail check we are able to update personal detail after passing module.
2. Check when update first name and last name, is name updating or not.
3. Check by updating all the required and optional fields of personal deatil form.
4. Check we are able to update the profile image of user or not.
5. While adding employee office detail check by providing correct or incorrect org id in body.
6. While adding employee office detail check by adding data in required and optional fields.
7. While adding employee office detail check the impact by not adding data in required fields.
8. While adding employee office detail check by not adding data in optional fields.
9. Check if by passing official email unique hrms id is generated for that user od not.
0. Check if the positions belongs to that specific department or not.
1. Check what happens if we cange the email and then again add the previous official email.
2. Check The validations of min and max salary.
3. Check if employee code is auto generated or not.
4. Check the sorting iorder on employees.
5. Verify if checks are handeled on hiring date or leaving date.
6. Check charcter length on hiring comment and leaving comment fields.

HEADERS

Authorization	{{bearerToken}}

Body formdata

organization	8
module	office
official_email	asma5@mailinator.com

Employee Bank Detail

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Bank Name Router

This router is used ot add bank names. Bank name and it's short code is added.

POST Post Bank Name

```
{{Url}}/api/banks/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "name": "Bank of Punjab",
    "short_form": "BOP"
}
```

GET List Bank Name

```
{{Url}}/api/banks/
```

HEADERS

Authorization	{{bearerToken}}

POST Post Bank Detial

```
{{Url}}/api/employee/{{empUUID}}/banks/details/
```

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

1. While adding employee bank detail check by providing correct or incorrect employee uuid in URL.
2. While adding employee bank detail check by adding data in required and optional fields.
3. While adding employee bank detail check the impact by not adding data in required fields.
4. While adding employee bank detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like bank.
6. Check the format and all validations on account number and IBAN number.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "bank": 1,
    "branch_name": "Lahore",
    "account_no": "222232-232",
    "account_title": "asma"
}
```

GET List Bank Detail

```
{{Url}}/api/employee/{{empUUID}}/banks/details/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee bank detail.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Bank Detail

GET Get specific Bank Detail

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

In URL pass Employee UUID and bank detail id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

DELETE Delete Bank Detail

```
{{Url}}/api/employee/{{empUUID}}/banks/details/14/
```

In URL pass Employee UUID and bank detail id.

Cases:

1. Check we are able to delete employee bank detail by providing it's id.
2. Check admin is able to delete employee bank detail of some other organization or not.
3. Check what happens if we delete deleted employee.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Bank Detail

```
{{Url}}/api/employee/{{empUUID}}/banks/details/14/
```

In URL pass Employee UUID and bank detail id.

Cases:

1. While updating employee bank detail check by providing correct or incorrect employee uuid in URL.

2. While updating employee bank detail check by adding data in required and optional fields.
3. While updating employee bank detail check the impact by not adding data in required fields.
4. While updating employee bank detail check by not adding data in optional fields.

7. Check the format and all validations on account number and IBAN number.
8. Check if we are able to active inactive bank detail.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "is_active": "true"
}
```

Employee Work Experience

Employee work experience APIS are used to add employee previous work experience. company name is added in a router and in work experience API that router is used to select company name. In URL employee UUID is passed that is created when we post employee personal detail data.

Company Name Router

This router is used to add Company names.

POST Post Company Name

```
{{Url}}/api/companies/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

Body raw (json)


```
{
    "name": "devsinc"
}
```

GET List Company Name

```
{{Url}}/api/companies/
```

HEADERS

Authorization	{{bearerToken}}

POST Post Company Detail

```
{{Url}}/api/employee/27d36957-48af-42b6-bf25-834b751e46e8/companies/
```

Cases:

1. While adding employee company detail check by providing correct or incorrect employee uuid in URL.
2. While adding employee company detail check by adding data in required and optional fields.
3. While adding employee company detail check the impact by not adding data in required fields.
4. While adding employee company detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like company.
6. Check the format of date field.
7. Check all the formats in upload experience letter field.

HEADERS

Authorization	{{bearerToken}}

Body formdata

company	1
designation	SQA

joining_date	2021-10-03

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

GET List Company Detail

```
{{Url}}/api/employee/27d36957-48af-42b6-bf25-834b751e46e8/companies/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee work experience.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Company Detail

```
{{Url}}/api/employee/{{empUUID}}/companies/40/
```

In URL pass Employee UUID and company detail id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

DELETE Delete Company Detail

```
{{Url}}/api/employee/{{empUUID}}/companies/38/
```

In URL pass Employee UUID and company detail id.

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

1. Check we are able to delete employee company detail by providing it's id.
2. Check admin is able to delete employee company detail of some other organization or not.
3. Check what happens if we delete deleted employee.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "employee": 296
}
```

PATCH Patch Company Detail

```
{{Url}}/api/employee/282d7ef7-5801-475a-a096-aec8ce5a5ae7/companies/6/
```

In URL pass Employee UUID and company detail id.

Cases:

1. While updating employee work experience detail check by providing correct or incorrect employee uuid in URL.
2. While updating employee work experience detail check by adding data in required and optional fields.
3. While updating employee work experience detail check the impact by not adding data in required fields.
4. While updating employee work experience detail check by not adding data in optional fields.
5. While adding employee work experience detail check by not adding data in optional fields.
6. Check by providing correct data in choise fields like company.
7. Check the format of date field.
8. Check all the formats in upload experience letter field.
9. If experience letter is added in post request while updating check by removing letter.
0. Check if we are able to active inactive company detail.

HEADERS

Authorization	{{bearerToken}}

Body formdata

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

is_active	True
company_name	SQA update
leaving_reason	dfdas drwds dcs
experience_letter	

Employee Contact Detail

Employee contact detail APIS are used to add employee contacty detail if in case of emergency empoloyee is not available there is contact of some one to whom company can contact. contact relation are added in a router and in contact detail API that router is used to select contact relation. In URL employee UUID is passed that is created when we post employee personal detail data.

Contact Relation Router

This router is used to add contact relations and data added in contact detail is added against these relations.

POST Post Conatct Relation

```
{{Url}}/api/employees/contact/relations/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "relation": "Friend"
}
```

GET List Conatct Relation

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

HEADERS

Authorization	{{bearerToken}}

POST Post Contact Detail

```
{{Url}}/api/employees/27d36957-48af-42b6-bf25-834b751e46e8/emergency/contact/
```

Cases:

1. While adding employee contact detail check by providing correct or incorrect employee uuid in URL.
2. While adding employee contact detail check by adding data in required and optional fields.
3. While adding employee contact detail check the impact by not adding data in required fields.
4. While adding employee contact detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like company.
6. Check the format and all validations on mobile number and email address field.

HEADERS

Authorization	{{bearerToken}}

PARAMS

Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjY5OTg4ODU3LCJpYXQiOjE2Njk4MTYwNTcsImp0aSI6ImU2NWU5MmZmYzRlZTQxNTJhMjc0MzNmMGRmYWZiMTRhIiwidXNlcl9pZCI6Mn0.u5rjTsLC1i69wQJvWEG3kFxRVdwS_do3FDcLoF7aSzo

Body raw (json)

json

```
{
    "relation": 1,
    "name": "asma",
    "mobile_no": "0303-4156775"
```

```
"mobile_no": "0303-4136775",
"address": "dha",
```


This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

GET List Contact Detail

```
{{Url}}/api/employees/27d36957-48af-42b6-bf25-834b751e46e8/emergency/contact/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee contact detail.

HEADERS

Authorization	{{bearerToken}}

PARAMS

GET Get specific Contact Detail

```
{{Url}}/api/employees/27d36957-48af-42b6-bf25-834b751e46e8/emergency/contact/47/
```

In URL pass Employee UUID and contact detail id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

PARAMS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

DELETE Delete Contact Detail

```
{{Url}}/api/employees/{{empUUID}}/emergency/contact/20/
```

In URL pass Employee UUID and contact detail id.

Cases:

1. Check we are able to delete employee contact detail by providing it's id.
2. Check admin is able to delete employee contact detail of some other organization or not.
3. Check what happens if we delete deleted employee contact detail.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Contact Detail

```
{{Url}}/api/employees/27d36957-48af-42b6-bf25-834b751e46e8/emergency/contact/31/
```

In URL pass Employee UUID and contact detail id.

Cases:

1. While updating employee contact detail check by providing correct or incorrect employee uuid in URL.
2. While updating employee contact detail check by adding data in required and optional fields.
3. While updating employee contact detail check the impact by not adding data in required fields.
4. While updating employee contact detail check by not adding data in optional fields.
5. While adding employee contact detail check by not adding data in optional fields.
6. Check by providing correct data in choise fields like relation.
7. Check the format and all validations on mobile number and email address field.
8. Check if we are able to active inactive contact detail.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

```
{
        "is_active": "true",
        "relation": 2
}
```

Employee Dependent Detail

Employee dependent detail APIS are used to add employee dependent detail. dependents are added in a router and in dependent detail API that router is used to select dependent. In URL employee UUID is passed that is created when we post employee personal detail data.

Dependent Router

This router is used to add dependent names. Dependent are parents, spouse and children.

POST Post Dependent

```
{{Url}}/api/employee/dependents/
```

GET Get Dependent

POST Post Dependent Detail

```
{{Url}}/api/employees/dependent/contact/
```

Cases:

1. While adding employee contact detail check by providing correct or incorrect employee uuid in URL.
2. While adding employee contact detail check by adding data in required and optional fields.
3. While adding employee contact detail check the impact by not adding data in required fields.
4. While adding employee contact detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like relationship.

HEADERS

Body raw (json)

```
json

{
    "relationship": 2,
    "name": "mother"
}
```

GET List Dependent Detail

```
{{Url}}/api/employees/{{empUUID}}/dependent/contact/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee dependent detail.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Dependent Detail

```
{{Url}}/api/employees/{{empUUID}}/dependent/contact/22/
```

In URL pass Employee UUID and dependent detail id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

DELETE Delete Dependent Detail

```
{{Url}}/api/employees/{{empUUID}}/dependent/contact/20/
```

In URL pass Employee UUID and dependent detail id.

Cases:

1. Check we are able to delete employee dependent detail by providing it's id.
2. Check admin is able to delete employee dependent detail of some other organization or not.
3. Check what happens if we delete deleted employee dependent detail.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Dependent Detail

```
{{Url}}/api/employees/{{empUUID}}/dependent/contact/20/
```

In URL pass Employee UUID and dependent detail id.

Cases:

1. While updating employee dependent detail check by providing correct or incorrect employee uuid in URL.
2. While updating employee dependent detail check by adding data in required and optional fields.
3. While updating employee dependent detail check the impact by not adding data in required fields.
4. While updating employee dependent detail check by not adding data in optional fields.
5. While adding employee dependent detail check by not adding data in optional fields.
6. Check by providing correct data in choise fields like relation.
7. Check if we are able to active inactive dependent detail.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```
{
    "is_active": "true"
}
```

Employee Education Detail

Employee education APIS are used to add employee education detail. Degree type and Instiute names are added in a router and in institute API the routers are used to select degree type and institute name. In URL employee UUID is passed that is created when we post employee personal detail data.While adding employee education detail, education certificates should also be added.

Degree Type Router

This router is used to add Degree type. Degree types are BS, MS etc.

POST Post Degree Type Router

```
{{Url}}/api/employees/degree/types/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title": "Masters",
    "duration": 2
}
```

GET List Degree Type Router

```
{{Url}}/api/employees/degree/types/
```

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Authorization {{bearerToken}}

Institute Name Router

This router is used to add Institute name.

POST Post Institue Name

```
{{Url}}/api/employees/institutes/
```

HEADERS

Authorization {{bearerToken}}

Body raw (json)

```json
{
"name": "Quaid I Azam University"
}
```

GET List Institute Name

```
{{Url}}/api/employees/institutes/
```

Cases:

1. While adding employee education detail check by providing correct or incorrect employee uuid in URL.
2. While adding employee education detail check by adding data in required and optional fields.
3. While adding employee education detail check the impact by not adding data in required fields.
4. While adding employee education detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like degree type or institue name.
6. Check the format of date field

6. Check the format of date field.
7. Check all the formats in upload degree certificate.

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Authorization	{{bearerToken}}

POST Post Education Detail

```
{{Url}}/api/emp/27d36957-48af-42b6-bf25-834b751e46e8/institutes/
```

HEADERS

Authorization	{{bearerToken}}

Body formdata

employee	33c1a51f-c82d-4b2d-8bdc-140b8e8fa1df
degree_title	BSSE
duration	4
year_of_completion	2017-07-30
degree_type	1
institue_name	PUCIT
degree_certificate	

GET List Education Detail

```
{{Url}}/api/emp/{{empUUID}}/institutes/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee education detail.

HEADERS

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

GET Get specific Education Detail

```
{{Url}}/api/emp/27d36957-48af-42b6-bf25-834b751e46e8/institutes/28/
```

In URL pass Employee UUID and education detail id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

PARAMS

DELETE Delete Education Detail

```
{{Url}}/api/emp/{{empUUID}}/institutes/16/
```

In URL pass Employee UUID and education detail id.

Cases:

1. Check we are able to delete employee education detail by providing it's id.
2. Check admin is able to delete employee education detail of some other organization or not.
3. Check what happens if we delete deleted employee education detail.

HEADERS

Authorization	{{bearerToken}}

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

```
{{Url}}/api/emp/27d36957-48af-42b6-bf25-834b751e46e8/institutes/29/
```

In URL pass Employee UUID and education detail id.

Cases:

1. While updating employee education detail check by providing correct or incorrect employee uuid in URL.
2. While updating employee education detail check by adding data in required and optional fields.
3. While updating employee education detail check the impact by not adding data in required fields.
4. While updating employee education detail check by not adding data in optional fields.
5. While adding employee education detail check by not adding data in optional fields.
6. Check by providing correct data in choise fields like degree type and institute name.
7. Check the format of date field.
8. Check all the formats in upload degree certificate field.
9. If degree certificate is added in post request while updating check by removing certificate.
0. Check if we are able to active inactive education detail.

HEADERS

Authorization	{{bearerToken}}

Body formdata

employee	296
is_active	True
institute_name	PUCIT update
degree_certificate	

GET Institute pre-data

```
{{Url}}/api/emp/pre/institutes/data/details/
```

HEADERS

Authorization	{{bearerToken}}

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Employee skills APIS are used to add employee skills detail. Skill name are added in a router and in skill detail API that router is used to select skill name. In URL employee UUID is passed that is created when we post employee personal detail data. While adding skill data proficiency level is added against each skill.

Skills Router

Skill router is used to add skills. Skills are added against catagories that are defined in above router.

Skill Catagory Router

StartFragment

Skill catagory router is used to define catagory of the skill. In this router only catagories are defined these catagories are used in skill router to define the skills. Catagories will be SQA, Web development, communuication etc.

EndFragment

POST Post Skill Catagory

```
{{Url}}/api/skills/categories/
```

HEADERS

Authorization	{{bearerToken}}

PARAMS

Body raw (json)

```
json
```

```
}
```

GET List Skill Catagory

```
{{Url}}/api/skills/categories/
```

HEADERS

Authorization	{{bearerToken}}

PARAMS

POST Post Skills

```
{{Url}}/api/skills/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "category": 2,
    "title": "good knowledge of agile methodologies"
}
```

GET List Skills

```
http://127.0.0.1:8000/api/skills/
```

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjcwNDk0MjI0LCJpYXQiOjE2NzAzMjE0MjQsImp0aSI6ImUyYzBiYTg5NTY4MTRkZjM4MTJmN2JjNDFjYTc1YmU1IiwidXNlcl9pZCI6Mn0.JeFpW_fuyqYQAejejR71Lm6msJFiYh2gOo2TPo8QW-Y

POST Post Skills Details

```
{{Url}}/api/employee/{{empUUID}}/skills/
```

Cases:

1. While adding employee skills detail check by providing correct or incorrect employee uuid in URL.
2. While adding employee skills detail check by adding data in required and optional fields.
3. While adding employee skills detail check the impact by not adding data in required fields.
4. While adding employee skills detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like skill or proficiency level.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "skill": 1,
    "proficiency_level": 2

}
```

GET List Skills Details

```
{{Url}}/api/employee/{{empUUID}}/skills/
```

Cases:

3. Verify all the data in response body is same that u added while adding employee skills detail.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Skills Details

```
{{Url}}/api/employee/{{empUUID}}/skills/15/
```

In URL pass Employee UUID and skill detail id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

DELETE Delete Skills Details

```
{{Url}}/api/employee/{{empUUID}}/skills/13/
```

In URL pass Employee UUID and skillsd detail id.

Cases:

1. Check we are able to delete employee skills detail by providing it's id.
2. Check admin is able to delete employee skills detail of some other organization or not.
3. Check what happens if we delete deleted employee skills detail.

HEADERS

Authorization	{{bearerToken}}

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

json

```
{
    "employee": 296
}
```

PATCH Update Skills Details

```
{{Url}}/api/employee/{{empUUID}}/skills/13/
```

In URL pass Employee UUID and skill detail id.

Cases:

1. While updating employee skill detail check by providing correct or incorrect employee uuid in URL.
2. While updating employee skill detail check by adding data in required and optional fields.
3. While updating employee skill detail check the impact by not adding data in required fields.
4. While updating employee skill detail check by not adding data in optional fields.
5. While adding employee skill detail check by not adding data in optional fields.
6. Check by providing correct data in choise fields like skill and proficiency level.
7. Check if we are able to active inactive skill detail.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "is_active": "true"
}
```

GET Skills Pre-date

```
{{Url}}/api/employee/pre/skills/data/details/
```

```
{{Url}}/api/employee/pie/skills/data/details/
```

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Authorization	{{bearerToken}}

Employee Project Roles

Employee project roles APIS are used to assign projects and roles to a project. Projecta=s and roles are added using projects and roles APIS and those project and roles are used in the empoloyee section to assign projects and roles of employee in that project. When a role is assigned to a employee its start date ios automatically added and when a new role is assigned to employee against that same project a new entry is created and end date is automatically added for the previous role.

Roles

User roles are very important part of a website. User are the people and roles are their functions. Roles are very important because on the basis of roles permessions for a user is defined. we have system roles and generic roles taht are assigned to a user against their projects.

Role Type Router

This router is used to add role type. There are 2 role types: Project roles and corporate roles.

POST Post Role Type

GET Get Role Type

POST Create Role

```
{{Url}}/api/roles/
```

Cases:

1. Check if roles are organization based .
2. While adding role check by providing org id in body.
3. While adding role check by adding data in required and optional fields.
4. While adding role check the impact by not adding data in required fields

5. While adding role check by not adding data in optional fields.
6. While adding roel check by adding same role code that already exist.

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

9. While adding role check by correct role type.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "role_type": 9,
    "title": "test",
    "code": 123
}
```

GET List Roles

```
{{Url}}/api/roles/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while creating role.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Role

```
{{Url}}/api/roles/60ccc4ca-8a1e-485c-a6d4-7c05798d111b/
```

In URL pass role UUID.

Cases:

1. View a specific role by providing inactive id.

4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

PARAMS

DELETE Delete Role

```
{{Url}}/api/roles/60ccc4ca-8a1e-485c-a6d4-7c05798d111b/
```

In URL pass role UUID.

Cases:

1. Check we are able to delete role by providing it's id.
2. Check admin is able to delete role of some other organization or not.
3. Check what happens if we delete deleted role.
4. Check if we are able to delete the role that is assigned to an employee.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Role

```
{{Url}}/api/roles/60ccc4ca-8a1e-485c-a6d4-7c05798d111b/
```

In URL pass role UUID.

Cases:

1. Check if roles are organization based .
2. While adding role check by providing org id in body.

3. While adding role check by adding data in required and optional fields.
4. While adding role check the impact by not adding data in required fields.

7. While adding role check by incorrect role type.
8. While adding role check by role type that does not exist.
9. While adding role check by correct role type.
0. check if we are able to active inactive role in update.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "is_active": "true"
}
```

Projects

To keep track of all the projects on which employees of the company work, projects are added.

POST Post Project

```
{{Url}}/api/projects/
```

StartFragment

Cases:

1. Check if projects are organization based .
2. While adding project check by providing org id in body.
3. While adding project check by adding data in required and optional fields.
4. While adding project check the impact by not adding data in required fields.
5. While adding project check by not adding data in optional fields.
6. While adding project check by adding same project code that already exist.

EndFragment

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Body raw (json)

```json
{
    "name": "HRMS",
    "code": "1223"
}
```

GET List Projects

```
{{Url}}/api/projects/
```

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding project.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Project

```
{{Url}}/api/projects/724d76ae-0bbd-44a5-b892-66de4b7c4f66/
```

In URL pass project UUID.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

PARAMS

DELETE Delete Project

```
{{Url}}/api/projects/724d76ae-0bbd-44a5-b892-66de4b7c4f66/
```

In URL pass Project UUID.

Cases:

1. Check we are able to delete project by providing it's id.
2. Check admin is able to delete project of some other organization or not.
3. Check what happens if we delete deleted project.
4. Check if we are to delete a project that is assigned to an employee.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Project

```
{{Url}}/api/projects/724d76ae-0bbd-44a5-b892-66de4b7c4f66/
```

In URL pass project UUID.

Cases:

1. Check if projects are organization based .
2. While adding projects check by providing org id in body.
3. While adding projects check by adding data in required and optional fields.
4. While adding projects check the impact by not adding data in required fields.
5. While adding projects check by not adding data in optional fields.
6. While adding projects check by adding same project code that already exist.
7. Check if we are able to active deleted role or not.

HEADERS

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Body raw (json)

```
json

{
    "is_active": "True"
}
```

GET Project Pre-data

```
{{Url}}/api/projects/pre/data/view/
```

HEADERS

Authorization	{{bearerToken}}

POST Post Project Roles

```
{{Url}}/api/employees/projects/roles/data/
```

Cases:

1. While adding employee project role detail check by providing correct or incorrect employee id in body.
2. While adding employee project role detail check by adding data in required and optional fields.
3. While adding employee project role detail check the impact by not adding data in required fields.
4. While adding employee project role detail check by not adding data in optional fields.
5. Check by providing correct data in choise fields like project and role.
6. When add data in response body check start date is automatically assigned to the employee or not.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```
{
    "employee": 165,
    "project": 9,
    "role": 1
}
```

GET List Project Roles

```
{{Url}}/api/employees/27d36957-48af-42b6-bf25-834b751e46e8/projects/roles/data/
```

In URL pass employeee UUID.

Cases:

1. Check if all the active data is appearing in get all API.
2. Check if in-active data is also appearing in get all API.
3. Verify all the data in response body is same that u added while adding employee project role detail.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Project Roles

```
{{Url}}/api/employees/projects/roles/37/data/
```

In URL pass Employee project role id.

Cases:

1. View a specific employee by providing inactive id.
2. View a specific employee by providing existing id.
3. View a specific employee by providing the id that doesnot exist.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

```
{{Url}}/api/employees/projects/roles/22/data/
```

In URL pass role id of employee project role.

Cases:

1. Check we are able to delete employee role by providing it's id.
2. Check admin is able to delete employee role of some other organization or not.
3. Check we are able to delete employee project by providing it's id.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Project Roles

```
{{Url}}/api/employees/projects/roles/37/data/
```

Cases:

1. While updating employee project role detail check by providing correct or incorrect employee uuid in URL.
2. While updating employee project role detail check by adding data in required and optional fields.
3. While updating employee project role detail check the impact by not adding data in required fields.
4. While updating employee project role detail check by not adding data in optional fields.
5. Check on providing project it shows proper error msg or not because we are not allowed to update project.
6. Check if we are able to update role by providing correct or incorrect role id.
7. Check if on updating role it create a new entry or mark previous role as completed by adding end date in it.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "role": 2
```

```
}
```

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

```
{{Url}}/api/employees/pre/projects/roles/data/view
```

In project role pre data API, pre-data of projects and role is appearing.

HEADERS

Authorization	{{bearerToken}}

DELETE Delete

```
http://127.0.0.1:8000/api/employees/projects/98/roles/data/
```

HEADERS

Authorization	{{BearerToken}}

Body raw (json)

```json
{
    "employee": 9,
    "project": 2,
    "role": 1
}
```

Employee Pre-data

Pre-data APIS are used to get all the data. In employee pre data API above mentioned all details of employees exist.

GET GET ALL

```
{{Url}}/api/employees/pre/complete/data/27d36957-48af-42b6-bf25-834b751e46e8/
```

HEADERS

This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

Authorization	{{bearerToken}}