Recruitment and Hiring

Instead of adding all JD data in job, we are adding JDs seperately and mantaining it as JD templates. In job we select the specific JD template and create job. Candidate can apply on that job. User can also view candidates against a job. After applying for a job candidates have to attempt an assesment test. Scoring against that assesment test will be done using complexity level. When candidate apply a stage is assigned to the candidate. User can also qualify or dis qualify the candidate at any stage by givibg the reason for dis qualifying. User can send email on each stage to candidate. Email templates are defined and emails are sent using those templates or user can also edit email template while sending. Also interview for a candidate can be set against a specific stage and evaluation against that interview will be submited by the interviewer.

JD description

For maintaining JD templates these APIS are used to add job description and data in it's dimensions.

JD dimension

This router is used to add JD dimensions. JD dimensions are added against JD types. We have total 7 JD dimensions. JD dimensions are level based.

JD types

This router is used to add JDtype. There are 2 JD types: Job specification and additional information.

POST POST JD type

```
{{Url}}/api/jd/types/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

PARAMS

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

Body raw (json)

```json
{
    "title": "Additional information",
    "level": 1
}
```

GET List JD type

```
{{Url}}/api/jd/types
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

GET Get Specific JD type

```
{{Url}}/api/jd/types/1/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

DELETE Delete JD type

```
{{Url}}/api/jd/types/4/
```

HEADERS

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

Accept	application/json

PARAMS

PATCH Update JD type

```
{{Url}}/api/jd/types/1/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

Body raw (json)

```json
{
    "title": "JD Specifications",
    "level": 1
}
```

POST Post JD dimension

```
{{Url}}/api/jd/dimensions/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

PARAMS

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

Body raw (json)

```json
{
    "title": "Work Environment",
    "level": 7,
    "jd_type": 2
}
```

GET List JD dimension

```
{{Url}}/api/jd/dimensions/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

GET Get specific JD dimension

```
{{Url}}/api/job/description/type/1/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

DELETE Delete JD dimension

```
{{Url}}/api/jd/dimensions/1/
```

HEADERS

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

Authorization	{{bearerToken}}
Accept	application/json

PATCH Update JD Dimension

```
{{Url}}/api/jd/dimensions/1/
```

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

PARAMS

Body raw (json)

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

POST Psot JD

```
{{Url}}/api/jd/
```

Cases:

1. While adding JD check by adding grouphead,SC and department that doesnot exist.
2. While adding JD check by adding grouphead,SC and department that is inactive
3. While adding JD check by adding grouphead,SC and department that belongs to some other organization.
4. While adding JD check by providing correct grouphead,SC and department id.
5. While adding JD check by providing org id in body.

6. While adding JD check by adding data in required and optional fields.
7. While adding JD check the impact by not adding data in required fields.
8. While adding JD check by not adding data in optional fields.

1. While adding JD check by adding data in array of job specification.

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

Body raw (json)

json


```
{
            "grouphead": 34,
            "department": 34,
            "project": "",
            "staff_classification": 26,
            "position": 24,
            "title": "additional info",
            "code": "JKHG",
            "main_responsibilities": "<p>descripotion</p>",
            "jd_specifications": [
                {
                    "jd_dimension": 1,
                    "essential": "Behavior  1. Ability and eagerness to learn new tools and tec

                }
            ]
}
```


GET List all JDs

```
{{Url}}/api/jd/
```

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 JD.

HEADERS

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

Accept	application/json

GET Get specific JD

```
{{Url}}/api/jd/3/
```

In URL pass JD id.

Cases:

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

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

DELETE Delete JD

```
{{Url}}/api/jd/13/
```

In URL pass JD id.

Cases:

1. Check we are able to delete JD by providing it's id.
2. Check admin is able to delete JD of some other organization or not.
3. Check what happens if we delete deleted JD.
4. Check if we are able to delete the JD against which we have an active job.

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

PARAMS

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

PATCH Update JD

```
{{Url}}/api/jd/3/
```

In URL pass position id.

Cases:

1. check if we are allowed to update JD by providing id that doesnot exist.
2. check if we are allowed to update inactive JD.
3. While adding JD check by adding organiztion that is inactive
4. While adding JD check by adding GH, SC and department that belongs to some other organization.
5. While adding JD check by providing correct GH,SC and department id.
6. While updating JD provide department id that doesnot belongs to that specific GH.
7. While adding JD check by adding data in required and optional fields.
8. While adding JD check the impact by not adding data in required fields.
9. Check character length of description field.
0. While adding Position check by adding same position code that already exist.
1. While adding JD check by updating jD dimensions.
2. Check by passing org id in request body.

HEADERS

Authorization	{{bearerToken}}
Accept	application/json

PARAMS

Body raw (json)

json

```
{
    "grouphead": 3,
    "grouphead_title": "GH update"
```

```
    "groupHead_title": "gh update",
    "department": 3,
    "project": "",
    "department_title": "Dept update",
```



```
    "position": 2,
    "position_title": "Position update",
    "title": "JD update",
    "code": "qwr5w35",
    "main_responsibilities": "<p>descripotion</p>",
    "jd_specifications": [
        {
            "jd": 3,
            "jd_dimension": 1,
            "jd_dimension_title": "Qualification",
            "essential": "asd",
            "desirable": "gh"
        },
        {
            "jd": 3,
            "jd_dimension": 2,
            "essential": "asd1",
            "desirable": "gh1"
        },
        {
            "jd": 3,
            "jd_dimension": 5,
            "essential": "asd2",
            "desirable": "gh2"
        }
    ]
}
```


Jobs

Following APIS are used to add job in the system. Job is added against job type. Candidate can apply against a job. Job can be deleted once it's purpose is fulfilled. and when we want we can again active the job to use it.

Job types

This router is used to add job type. There are 2 job types: permanent and contract.

POST Post job type

```
{{Url}}/api/job/types/
```

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

{
    "title": "contract",
    "level": 2

}
```

GET List job type

```
{{Url}}/api/job/types/
```

HEADERS

Authorization	{{bearerToken}}

GET Get specific job type

```
{{Url}}/api/jobs/
```

HEADERS

Authorization	{{bearerToken}}

DELETE Delete job type

```
{{Url}}/api/job/types/3/
```

PATCH Update job type

```
{{Url}}/api/job/types/2/
```

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

Authorization {{bearerToken}}

Body raw (json)

json

```
{
    "level": 2
}
```

POST Create Job

```
{{Url}}/api/jobs/
```

Cases:

1. While adding job check by adding grouphead,SC and department that doesnot exist.
2. While adding job check by adding grouphead,SC and department that is inactive
3. While adding job check by adding grouphead,SC and department that belongs to some other organization.
4. While adding job check by providing correct grouphead,SC and department id.
5. While adding job check by providing org id in body.
6. While adding job check by adding data in required and optional fields.
7. While adding job check the impact by not adding data in required fields.
8. While adding job check by not adding data in optional fields.
9. While adding job check by adding same job code that already exist.
0. In JD_selection field provide id that does not exist.

HEADERS

Authorization {{bearerToken}}

Body raw (json)

json

```
{
"department": 4
```

```
"department": 4,
"jd_selection": 6,
"job_code": "9995",
"job_type": 1,
```



```
"staff_classification": 4,
"title": "test delete"
}
```


GET List Job

```
{{Url}}/api/jobs/
```

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 job.

HEADERS

Authorization	{{bearerToken}}

GET Get Specifi Job

```
{{Url}}/api/jobs/4ab61440-b9f2-4336-b1da-4fc1afea7b45/
```

In URL pass JD id.

Cases:

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

HEADERS

Authorization	{{bearerToken}}

DELETE Delete Job

```
{{Url}}/api/jobs/f4ac4195-bdc6-4936-8ede-50cbf9898052/deactivate/
```

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

In URL pass job UUID.

Cases:

- Check we are able to delete job by providing it's UUid.
- Check admin is able to delete job of some other organization or not.
- Check what happens if we delete deleted job.

EndFragment

PATCH Update Job

```
{{Url}}/api/jobs/8fe421e6-f23f-41c6-9d66-00c3cd263533/
```

In URL pass job UUID.

Cases:

1. check if we are allowed to update job by providing id that doesnot exist.
2. check if we are allowed to update inactive job.
3. While adding job check by adding organiztion that is inactive
4. While adding job check by adding GH, SC and department that belongs to some other organization.
5. While adding job check by providing correct GH,SC and department id.
6. While updating job provide department id that doesnot belongs to that specific GH.
7. While adding job check by adding data in required and optional fields.
8. While adding job check the impact by not adding data in required fields.
9. Check character length of description field.
0. While adding job check by adding same job code that already exist.
1. Check by passing org id in request body.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
"job_post_id": 17,
"department": 4,
```

```
"jd_selection": 6,
"job_code": "9997",
"job_type": 1,
```



```
"staff_classification": 4,
"title": "job2 update2"
}
```


PATCH Update Job Post

```
{{Url}}/api/jobs/3edcee30-537f-4b46-80f2-304833018dd1/eactivate/
```

This API is used to activate deleted job.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json


```
{
"department": 4,
"jd_selection": 6,
"job_code": "9995",
"job_type": 1,
"no_of_individuals": 5,
"position": 6,
"staff_classification": 4,
"title": "job update"
}
```


GET Job PreData

```
{{Url}}/api/jobs/pre/data/7/
```

This API is used to get all the data of all jobs.

HEADERS

Authorization	{{bearerToken}}

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

It contains APIS of assesment test. Assesment test is of two types: technical and non-technical. Technical test is added against positions. Each position will contain different technical test. Assesment test should be import in csv utf-8 format. The file should follow followong instructions:

- images are not allowed.
- answers should be in form of integer.column no(1-8)
- Complexty_level on scale of 1 to 5.
- total_options: no of options of qs (1 -8)

Assesment test all APIS are not in the list.

Assesment type

Assesment test is cataogrized into 2 types according to assement type router. There are 2 assesment test types: technical and non-technical.

POST Post assesment type

```
{{Url}}/api/assessment/types/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
        "title": "Non-Technical"
}
```

GET List assesment type

```
{{Url}}/api/assessment/types/
```

HEADERS

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

PATCH Update assesment type

```
{{Url}}/api/assessment/types/
```

HEADERS

Authorization	{{bearerToken}}

Candidate

This folder contains APIS to add candidate in the system againts a job. When a candidate is added by default applied stage is assigned to him. Stage can be alos be updated

POST Post Candidate

```
{{Url}}/api/candidates/apply/form/5c5bcc9f-29b5-4bba-a73b-68723a6eae78/
```

Candidate can be added using "job UUID" that is UUID of job post.

Cases:

1. While adding candidate check by passing job UUID in URL.
2. While adding candidate check by providing org id in body.
3. While adding candidate check by adding data in required and optional fields.
4. While adding candidate check the impact by not adding data in required fields.
5. While adding candidate check by not adding data in optional fields.
6. Check validation on CNIC number, mobile number and email address.
7. Check if a candidate can apply on same job with same CNIC number more then once.
8. Check If Using the already registered CNIC number candidate can apply for a new job or not.
9. Check the format of resume.
0. Check in response body if a stage is assigned to the candidate or not.
1. Check if candidate is_qualified is true or not.

HEADERS

Accept	application/json

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

candidate_name	"word file"
cnic_no	44444-4444444-9
email	asmazahid603@gmail.com
mobile_no	03034156776
resume	
stage	1

GET List all candidates

```
{{Url}}/api/candidates/
```

Used to get list of all the candidates in the system.

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.

HEADERS

Authorization	{{bearerToken}}

PATCH Update stage of candidate

```
{{Url}}/api/candidates/stage/update/
```

This API is used to update stage of a candidate. In request body pass the UUID of the candidate and the stage.

Cases:

1. While updating stage check by adding data in required and optional fields.
2. While updating stage check the impact by not adding data in required fields.
3. While updating stage check by not adding data in optional fields.

4. Check by passing wrong UUID.
5. Check by passing stage that doesnot exist.

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

Authorization {{bearerToken}}

Body raw (json)

```
json

{
    "uuid": "ca1a99a5-efe3-40a0-89ab-02c3a88364cc",
    "stage":16
}
```

Procedure Types

Procedure Type is a router in which we only pass title. Basically prodeure type is used in sending emails setting interviews and doing evaluations. For now we have only 1 procedure that is recruitment and selection.

POST Create

```
{{Url}}/api/organizations/procedures/types/
```

HEADERS

Authorization {{bearerToken}}

Body raw (json)

```
json

{
    "title": "Recruitment and Hiring"
}
```

GET List

```
{{Url}}/api/organizations/procedures/types/
```

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

Authorization {{bearerToken}}

GET Get

```
{{Url}}/api/organizations/procedures/types/1/
```

HEADERS

Authorization {{bearerToken}}

PATCH Update

```
{{Url}}/api/organizations/procedures/types/1/
```

HEADERS

Authorization {{bearerToken}}

Body raw (json)

json

```
{
    "title": "Job & Recruitment"
}
```

Stages

Stages are added against procedure and are added to assigned to the canddiates. Each stage has a level and on sinfge level only 1 stage is added. Whether email, interview or evaluation is set against an interview or not is defined while adding stage.

For now stage is acting as a router.

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

```
{{Url}}/api/stages/
```

Cases:

1. While adding stage check by adding data in required and optional fields.
2. While adding stage check the impact by not adding data in required fields.
3. While adding stage check by not adding data in optional fields.
4. While adding JD check by providing org id in body.
5. Check by providing correct/ incorrect id of procedure.
6. Check by adding more then 1 stage on a single level.
7. Check by adding same stage again.
8. Check the impact by setting is_email, is_interview and is_evaluation true or false.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title": "Offer Letter",
    "procedure": 5,
    "level":6,
    "is_interview": "false",
    "is_evaluation": "false",
    "is_email": "true"
}
```

GET List all stages

```
{{Url}}/api/stages/
```

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 stage.

HEADERS

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

DELETE Delete stage

```
{{Url}}/api/stages/6/
```

In URL pass stage id.

HEADERS

Authorization	{{bearerToken}}

PATCH Update stage

```
{{Url}}/api/stages/17/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "is_email": true
}
```

DisQualify/Qualify

This folder contain API to disqualify or qualify candidate. A status is passed on the base of which candiodate will be qualified or dis-qualified. 0 represent dis-qualify and 1 represent qualify. Also in request body we need to pass reason to qualify or dis-qualify the candidate.

POST Dis-Qualify candidate

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

In URL pass UUID of candidate.

Cases:

1. While changing status check by adding data in required and optional fields.
2. While changuing status the impact by not adding data in required fields.
3. While changing status check by not adding data in optional fields.
4. Check on passing 0 in status, in response body is_qualified becomes false or not.
5. Check the character length in reason field.
6. Check if after dis_qualifying candidate we can again qualify the candidate or not.
7. Check if after dis-qualifying candidate, we can still set candidate interview or send him the email or not.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "candidate_status": 0,
    "reason": "gcshbh bcsjh"
}
```

Email

This folder contain APIS for adding email variable, email templates and for sending the emails to the candidates.

Email Template

It contains APIS through which email template is set. While setting email tempalte title for each tempalte is given and subject body and footer for each email is given. In subject, body or footer email template variables are also used where required.

Email Template Variable

Email Template Variable

be set as email template variable. While adding email templalte variables, code for variable should always be added in square brackets with an at the rate sign like [@code] and it will be used as it is in the email template.

POST Post Variable

```
{{Url}}/api/email/template/variables/
```

Cases:

1. While adding email template variable check by adding data in required and optional fields.
2. While adding email template variable check the impact by not adding data in required fields.
3. While adding email template variable check by not adding data in optional fields.
4. While adding email variable check by providing org id in body.
5. Check by adding same title and code again.
6. Check by not following the code fomat.
7. Check character length of description field.
8. Check if is_email is set false cam that variable be used in email template.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title": "position_title",
    "code" : "[@position_title]",
    "short_code": "posT",
    "description" : "",
    "is_email" : "true"
}
```

GET List Variable

```
{{Url}}/api/email/template/variables/
```

Cases:

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

HEADERS

Authorization	{{bearerToken}}

PATCH Update Variable

```
{{Url}}/api/email/template/variables/6/
```

In URL pass email variable id.

Cases:

1. check if we are allowed to update email template variable by providing id that doesnot exist.
2. check if we are allowed to update inactive email template variable.
3. While adding email template variable check by adding data in required and optional fields.
4. While adding email template variable check the impact by not adding data in required fields.
5. While adding email template variable check by not adding data in optional fields.
6. While adding email variable check by providing org id in body.
7. Check by adding same title and code again.
8. Check by not following the code fomat.
9. Check character length of description field.
0. Check if is_email is set false can that variable be used in email template.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "code" : "[@company_name]"
}
```

POST Post Email Template

```
{{Url}}/api/email/templates/
```

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

1. While adding email template check by adding data in required and optional fields.
2. While adding email template check the impact by not adding data in required fields.
3. While adding email template check by not adding data in optional fields.
4. While adding email template check by providing org id in body.
5. Check by not following the fomat of email template variable.
6. Check character length of subject, body and footer field.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title" :"test template 1",
    "subject_line": "[@company_name]",
    "body": "Dear [@candidate_name], thanks for applying for the [@position_title] ",
    "footer": "best regards"
}
```

GET List Email Template

```
{{Url}}/api/email/templates/
```

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 email template.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Email Template

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

In URL pass email template UUID.

Cases:

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

HEADERS

Authorization	{{bearerToken}}

DELETE Delete Email Template

```
{{Url}}/api/email/templates/b77eea8e-ae9f-40cc-922d-03fd4386958f/
```

In URL pass email template UUID.

Cases:

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

HEADERS

Authorization	{{bearerToken}}

PATCH Update Email Template

```
{{Url}}/api/email/templates/29202b94-c091-4dc2-a6cc-8f6887fd7328/
```

In URL pass email template UUID.

Cases:

1. While updating email template check by adding data in required and optional fields.

2. While updating email template check the impact by not adding data in required fields.
3. While updating email template check by not adding data in optional fields.
4. While updating email template check by providing org id in body.

7. Check if we are able to active deleted template.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```
json

{
    "subject": "[@company_name]",
    "body": "Dear [@candidate_name], Hope this email finds you well. u applied for [@position_t
    "footer": "best regards"
}
```

Candidate email

After having email template this folder conatin all the APIS used to in sending the email to the candidate.

POST Set Email

```
{{Url}}/api/email/templates/candidate/job/ca1a99a5-efe3-40a0-89ab-02c3a88364cc/
```

StartFragment

This API is used to set email for a candidate. In URL send UUID of candidate named as "UUID". In request body pass the stage and select the email template for that stage.

Cases:

1. While setting email check by adding data in required and optional fields.
2. While setting email check the impact by not adding data in required fields.
3. While setting email check by not adding data in optional fields.
4. Check by passing wrong UUID in URL.
5. Check by passing stage that doesnot exist or is inactive.
6. Check by selecting email template that doesnot exist or is inactive.

EndFragment

HEADERS

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

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "stage": 17,
    "email_template": 9
}
```

GET View Specific email

```
{{Url}}/api/email/templates/candidate/job/view/ca1a99a5-efe3-40a0-89ab-02c3a88364cc/71/
```

This API is used to view the email that is set against the candidate.

Cases:

1. View data by providing inactive id that is set against the the candidate for email.
2. View data by providing existing id that is set against the the candidate for email.
3. View data by providing id that does not exist in DB that is set against the the candidate for email.
4. View all the data in repsonse body.

HEADERS

Authorization	{{bearerToken}}

PATCH Save that email

```
{{Url}}/api/email/templates/candidate/job/save/8d51d786-e133-42fc-ba71-2cb174e0b91d/63/
```

This API is used to save the email that we set to send to a candidate. In set we select a template and then in save API we make changes in that template if we want to make. In URL candidate UUID and id of set email is used.

Cases:

1. While saving email check by adding data in required and optional fields.
2. While saving email check the impact by not adding data in required fields.

2. While saving email check the impact by not adding data in required fields.
3. While saving email check by not adding data in optional fields.
4. While adding email template check by providing org id in body.

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

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
     "subject": "[@company_name]",
      "body": "<p>Hi [@candidate_name]</p>",
        "footer": "<p>Best regards</p>"
}
```

POST Reset email against a candidate

```
{{Url}}/api/email/templates/candidate/job/reset/stage/9238d283-95c3-4d95-85e9-ade075e84b2a/7/
```

Using this API user can change the email template to send to the candidate in case he select wrong email template befire sending he cam reset the email template. In URL candidate UUID and id of set email is used.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
        "email_template": 8
}
```

GET Send the email

```
{{Url}}/api/email/templates/candidate/job/email/send/8d51d786-e133-42fc-ba71-2cb174e0b91d/63/
```


template if we want to make and after that we press send to send the email to the candidate. In URL candidate UUID and id of set email is used.

HEADERS

Authorization	{{bearerToken}}

GET List emails against that candidate

```
{{Url}}/api/email/templates/candidate/job/f465a657-67aa-4fea-abc9-18396e19a3a2/
```

This API is used to view the email that is set against the candidate.

HEADERS

Authorization	{{bearerToken}}

GET Get against stage

```
{{Url}}/api/email/templates/candidate/job/get/stage/d768845e-0e29-4a37-a400-06fe17006c41/7/
```

This API is used to get email against a astage. In URL candidate UUID and stage is provided against which we want to get emails.

HEADERS

Authorization	{{bearerToken}}

Interview

Candidate interview can be set using system so that system can keep record how many total interviews are taken for a position. Interview time is set in the system and an email is sen t to the candidate for interview. Interview can only be set against stage which has is_interview TRUE.

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

Interview mode is a router in which we add the mode of interview whether interview is online or onsite.

POST Post Mode

```
{{Url}}/api/interviews/mode/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "title": "Onsite interview"
}
```

GET List Mode

```
{{Url}}/api/interviews/mode/
```

HEADERS

Authorization	{{bearerToken}}

PATCH Update Mode

```
{{Url}}/api/interviews/mode/
```

HEADERS

Authorization	{{bearerToken}}

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

```json
{
    "title": "online"
}
```

interview Time Slots

Interview time slot is used when user is scheduling the interview. While adding time slots time interval is apssed in URL and request body. Title of time slot should be like start time for interview and end time of interview.

Time Interval

Time interval is also a router used in time slots. In time interval title is passed that will be morning or evening. Start time and end time for that interval is defined

POST Post Interval

```
{{Url}}/api/time/intervals/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "title": "Morning",
    "level": 1,
    "start_time": "12:00",
    "end_time": "02:00"
}
```

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

```
{{Url}}/api/time/intervals/
```

HEADERS

Authorization	{{bearerToken}}

PATCH Update Interval

```
{{Url}}/api/time/intervals/1/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title": "morning",
    "level": 1,
    "start_time": "10:00",
    "end_time": "12:00"
}
```

POST Post Time Slot

```
{{Url}}/api/time/intervals/8/slots/
```

In URL pass time interval id.

Cases:

1. While adding time slot check by adding data in required and optional fields.
2. While adding time slot check the impact by not adding data in required fields.
3. While adding time slot check by not adding data in optional fields.

4. While adding time slot check by providing org id in body.
5. Check from time and to time is same as in title.

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

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "title": "01:00 - 02:00",
    "from_time":"01:00",
    "to_time": "02:00",
    "time_interval":8
}
```

GET List Time Slot

```
{{Url}}/api/time/intervals/3/slots/
```

In URL pass time interval id.

Cases:

1. Check if all the active data is appearing against a specific interval 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 tiem slots.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Time Slot

```
{{Url}}/api/time/intervals/2/slots/4/
```

In URL pass time interval and time slot id.

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

1. While updating time slot check by adding data in required and optional fields.
2. While updating time slot check the impact by not adding data in required fields.
3. While updating time slot check by not adding data in optional fields.
4. While updating time slot check by providing org id in body.
5. Check from time and to time is same as in title.
6. Check by providding inactve interval id in body.
7. Check if we are able to active deactive tiem slot.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "title": "05:30 - 06:30",
    "from_time":"05:30",
    "to_time": "06:30",
    "time_interval":2
}
```

Candidate Interview

It contains APIs used in the process of interview i.e: tto set interview, start, cancel, complete or reschedule interview.

POST Set Interview

```
{{Url}}/api/interviews/candidate/job/ca1a99a5-efe3-40a0-89ab-02c3a88364cc/
```

In URL pass UUID of candidate.

Cases:

1. While setting interview check by adding data in required and optional fields.
2. While setting interview check the impact by not adding data in required fields.
3. While setting interview check by not adding data in optional fields.
4. While setting interview check by providing org id in body.

5. Check by passing correct, incorrect id of interviewer.
6. Check by passing correct, incorrect id of stage.

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

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "interviewer": 28,
    "stage": 17
}
```

PATCH Update set Interview

```
{{Url}}/api/interviews/candidate/job/23cef10c-86ff-49c2-8436-983a4180de96/17/
```

This API is used to update the interview that user set. In URL pass UUID of candidate and id of set interview.

Cases:

1. While updating interview details check by adding data in required and optional fields.
2. While updating interview details check the impact by not adding data in required fields.
3. While updating interview details check by not adding data in optional fields.
4. While updating interview details check by providing org id in body.
5. Check by passing correct, incorrect id of interviewer.
6. Check by passing correct, incorrect id of stage.
7. Check by giving wrong date format of interview date.
8. Check by passing correct or incorrect id of interview mode.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json


```
    "interview_date": "2023-10-01",
    "interview_time_slot": 1,
    "interview_mode": 1
}
```

GET Start Interview

```
{{Url}}/api/interviews/candidate/job/start/f465a657-67aa-4fea-abc9-18396e19a3a2/24/
```

In URL pass UUID of candidate and id of interview that is set. It is a get request when hit start interview, interview will be started and in response body start date and time of interview will start appearing. Interview will be start after it is set or it is rescheduled. After cancel or completing interview start button should through error.

HEADERS

Authorization	{{bearerToken}}

PATCH Cancel Interview

```
{{Url}}/api/interviews/candidate/job/cancel/23cef10c-86ff-49c2-8436-983a4180de96/20/
```

In URL pass UUID of candidate and id of interview that is set. It is a patch request. When hit cancel interview reason to cancel the interview should also be sent in request body and in response body cancel date and time with reason of cancellation should appearing. Interview will be cancelled after it is set or after interview is started. After cancel interview need to set the interview again. After cancelling interview we can not start or complete the interview.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "reason_for_cancel": "check active status"
```

```
}
```

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

```
{{Url}}/api/interviews/candidate/job/reschedule/23cef10c-86ff-49c2-8436-983a4180de96/17/
```

In URL pass UUID of candidate and id of interview that is set. It is a post request. We can reschedule interview after it is cancelled or once interview is completed. Reschedule interview API works like Set interview API.

Cases:

1. While rescheduling interview check by adding data in required and optional fields.
2. While rescheduling interview check the impact by not adding data in required fields.
3. While rescheduling interview check by not adding data in optional fields.
4. While rescheduling interview check by providing org id in body.
5. Check by passing correct, incorrect id of interviewer.
6. Check by passing correct, incorrect id of stage.
7. Check by giving wrong date format of interview date.
8. Check by passing correct or incorrect id of interview mode.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```
json

{
    "interviewer": 17,
    "stage": 4
}
```

GET Complete Interview

```
{{Url}}/api/interviews/candidate/job/mark/complete/f465a657-67aa-4fea-abc9-18396e19a3a2/24/
```

In URL pass UUID of candidate and id of interview that is set. It is a get request when hit complete interview, interview will be completed and in response body complete date and time of interview will start appearing. Interview will be completed once interview is satrted.

HEADERS

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

GET Get Interview

```
{{Url}}/api/interviews/candidate/job/ca1a99a5-efe3-40a0-89ab-02c3a88364cc/37/
```

In URL pass UUID of candidate and id of interview that is set. Verify all the data in response body.

HEADERS

Authorization	{{bearerToken}}

Evaluation

After interview has been taken candidate evaluation forms are to be completed by the interviewer to rank the candidate's overall qualifications for the position to which they have applied. Evaluation can only be done if in case is_evaluation is set as TRUE while adding stages

Evaluation Questions

Evaluation questions are added against each evaluation type to evaluate the candidate. Against these questions candidate marks are calculated.

Evaluation type

it is a router in which types of evaluations will be set like Technical evaluation, HR evaluation etc. Whie adding evaluation type title of evaluation is given and each title must be added against procedure.

POST Post Evaluation Type

```
{{Url}}/api/evaluations/
```

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
{
    "title": "Final Evaluation",
    "description": "",
    "procedure": 4
}
```

GET List Evaluation Type

```
{{Url}}/api/evaluations/
```

HEADERS

Authorization	{{bearerToken}}

GET Get specific Evaluation Type

```
{{Url}}/api/evaluations/223ec049-beb5-469a-bd50-81472e0ccab2/
```

HEADERS

Authorization	{{bearerToken}}

PATCH Update Evaluation Type

```
{{Url}}/api/evaluations/319f7157-54f7-4009-8a38-833e5323eb2e/
```

HEADERS

Authorization	{{bearerToken}}

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

```json
{
    "procedure": 5
}
```

POST Post evaluation Qs

```
{{Url}}/api/evaluations/47ea87df-628e-4588-a628-0a324b70456e/procedure/questions/
```

In URL evaluation type UUID is send. complexity level should be between 1 to 5.

Cases:

1. While adding evaluiation question check by adding data in required and optional fields.
2. While adding evaluiation question check the impact by not adding data in required fields.
3. While adding evaluiation question check by not adding data in optional fields.
4. While adding evaluiation question check by providing org id in body.
5. Check by passing correct, incorrect id of evaluation type.
6. Check by passing correct, incorrect id of procedure.
7. Check by giving complexity level more then 1.
8. Check by adding same qs against same complexity level and same type more then once.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "question" : "Leadership skills",
    "complexity_level" : 4,
    "evaluation" : 9,
    "procedure": 5
}
```

GET List Evaluation Qs

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

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

HEADERS

Authorization	{{bearerToken}}

GET Get specific Qs

```
{{Url}}/api/evaluations/e263910f-f69f-430a-a0d2-0cef25894308/procedure/questions/5/
```

In URL evaluation type UUID and evaluation question is is send.

Cases:

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

HEADERS

Authorization	{{bearerToken}}

PATCH Update Evaluation Qs

```
{{Url}}/api/evaluations/36cff7bd-2697-410a-8f02-0704cc6c425a/procedure/questions/8/
```

In URL evaluation type UUID and evaluation question is is send. complexity level should be between 1 to 5.

Cases:

1. While updating evaluiation question check by adding data in required and optional fields.
2. While updating evaluiation question check the impact by not adding data in required fields.

3. While updating evaluiation question check by not adding data in optional fields.
4. While updating evaluiation question check by providing org id in body.
5. Check by passing correct, incorrect id of evaluation type.

8. Check by adding same qs against same complexity level and same type more then once.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```
json

{
    "question": "Time management"
}
```

Candidate Evaluation

StartFragment

It contains APIs used in the process of evaluation i.e: to set evaluation, start, cancel, complete recheck or markdone evaluation.

EndFragment

POST Post Set Evaluation

```
{{Url}}/api/evaluations/candidate/job/set/b2efda8a-718c-4c4b-a3f5-ad43b3375692/
```

In URL pass UUID of candidate.

Cases:

1. While setting evaluation check by adding data in required and optional fields.
2. While setting evaluation check the impact by not adding data in required fields.
3. While setting evaluation check by not adding data in optional fields.
4. While setting evaluation check by providing org id in body.
5. Check by passing correct, incorrect id of employee who have to done evaluation in case interview of the candidate is not done or set.
6. Check by passing correct, incorrect id of stage.
7. Check by passing correct, incorrect id of evaluation

7. Check by passing correct, incorrect id of evaluation.

HEADERS

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

Authorization	{{bearerToken}}

Body raw (json)

```
json

{
    "evaluation": 7,
    "stage": 4
}
```

GET List Evaluation

```
{{Url}}/api/evaluations/candidate/job/list/b3fca7ba-4149-4ba2-b854-e650bdcc9b26/
```

In URL pass UUID of candidate. Verify all the data in response body.

HEADERS

Authorization	{{bearerToken}}

PATCH Update Set Evaluation

```
{{Url}}/api/evaluations/candidate/job/update/23cef10c-86ff-49c2-8436-983a4180de96/7/
```

In URL pass UUID of candidate and id of evaluation that is set.

Cases:

1. While updating evaluation check by adding data in required and optional fields.
2. While updating evaluation check the impact by not adding data in required fields.
3. While updating evaluation check by not adding data in optional fields.
4. While updating evaluation check by providing org id in body.
5. Check by passing correct, incorrect id of employee who have to done evaluation in case interview of the candidate is not done or set.
6. Check by passing correct, incorrect id of stage.
7. Check by passing correct, incorrect id of evaluation.

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
{
    "evaluated_by": 17
}
```

GET Start Evaluation

```
{{Url}}/api/evaluations/candidate/job/start/b2efda8a-718c-4c4b-a3f5-ad43b3375692/10/
```

In URL pass UUID of candidate and id of evaluation that is set. It is a get request when hit start evaluation, evaluation will be started and in response body start date and time of evaluation will start appearing. Evaluation will be started after it is set. After cancel or completing evaluation, we should not be able to start an evaluation.

HEADERS

Authorization	{{bearerToken}}

PATCH Cancel Evaluation

```
{{Url}}/api/evaluations/candidate/job/cancel/b2efda8a-718c-4c4b-a3f5-ad43b3375692/9/
```

In URL pass UUID of candidate and id of evaluation that is set. It is a patch request. When hit cancel evaluation reason to cancel the evaluation should also be sent in request body and in response body cancel date and time with reason of cancellation should appearing. Interview will be cancelled after it is set or after interview is started. After cancel cancel need to set the evaluation again. After cancelling evaluation we can not start or complete the evaluation.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```
{
    "reason_for_cancel":"test cancel"
}
```

POST Complete Evaluation

```
{{Url}}/api/evaluations/candidate/job/submit/questions/remarks/b2efda8a-718c-4c4b-a3f5-ad43b3375692/10/
```

In URL pass UUID of candidate and id of evaluation that is set. It is a post request when hit complete evaluation after sending question id and employee marks, evaluation will be completed and in response body complete date and time of interview will start appearing. Evaluation will be completed once evaluation is satrted.

In evaluation question remarks array if of question and score is sent also in the request body recommendation and remarks are aso sent.

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

json

```
{
    "evaluation_questions_remarks": [
            {
                "id": 22,
                "score": 5
            },
            {
                 "id": 23,
                "score": 6
            }
    ],
    "recommendation": "not recomended",
    "comment": "xdf cf"
}
```

GET Rechecked Evaluation

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

In URL pass UUID of candidate and id of evaluation that is set. It is a get request when hit recheck evaluation after completing evaluation, evaluation will be cross verified and in response body recheck date and time of interview will start appearing. Evaluation will be rechecked once evaluation is completed.

HEADERS

Authorization	{{bearerToken}}

GET Markdone Evaluation

```
{{Url}}/api/evaluations/candidate/job/mark/done/questions/remarks/b2efda8a-718c-4c4b-a3f5-ad43b3375692/10/
```

In URL pass UUID of candidate and id of evaluation that is set. It is a get request when hit markdone evaluation after completing or re checking evaluation, evaluation will be completed.

HEADERS

Authorization	{{bearerToken}}

GET Get specific Evaluation

```
{{Url}}/api/evaluations/candidate/job/get/b3fca7ba-4149-4ba2-b854-e650bdcc9b26/8/
```

In URL pass UUID of candidate and id of evaluation. Verify all the data in response body.

HEADERS

Authorization	{{bearerToken}}

Scores

This forlder contains APIS of calcualting score of candidate assesment test and evaluation. In future these APIS are also used to calculate scores for training and evaluation of employee. Score is static taht is 5. Complexity level is added that is

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

Complexity Levels

This router is used to add all the complexity levels from 1 to 5 in the system. While adding complexity level add title code and level in post request.

POST Post Complexity Level

```
{{Url}}/scores/complexity/levels/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "title":"Complexity Level 5",
    "code": "cl5",
    "level": 5
}
```

GET List Complexity level

```
{{Url}}/scores/complexity/levels/
```

HEADERS

Authorization	{{bearerToken}}

PATCH Update Complexity Level

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

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "title": "CL 1",
    "code": "cl1"
}
```

Score Types

Score type is also a router used to define the type for which we are calculting score like for candidate assesment test, candidate evaluation, employee evaluation.

POST Post Score Type

```
{{Url}}/scores/types/
```

HEADERS

Authorization	{{bearerToken}}

Body raw (json)

```json
{
    "title": "Evaluation Score"
}
```

GET List Score Type

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

HEADERS

Authorization	{{bearerToken}}

GET Candidate pre-data

```
{{Url}}/api/candidate/job/get/stage/e6d78a1e-334b-41b6-9265-d5e45a78f4ee/5/
```