Integrations Developer Remote Assessment

The goal of this assessment project is to develop a simple RESTful API to retrieve and display data residing in an Oracle database. The API should match the format requested below. The assessment will be judged on the understanding of the requirements presented plus code quality and adherence to best practices. Where time or misunderstanding does not allow for completion, please document this and include it with your submission. Communication skills are also considered when choosing the best candidate for the role.

Requirements

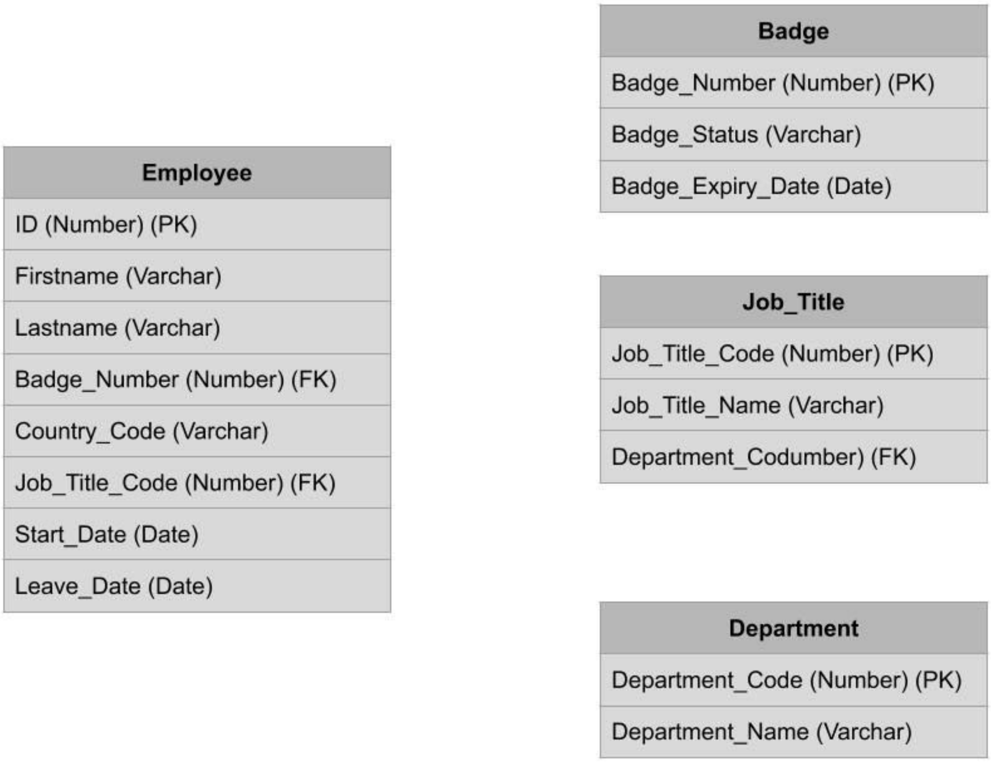
- Develop a RESTful API with endpoints matching the below format
- Build API using one of Java, NodeJS or Python
- API to use basic authentication
- API to return JSON Arrays in all successful responses
- Submission to include a github link or a zip file containing all relevant parts
- Include simple document outlining steps to run application locally
- Include simple document, if applicable, outlining any issues with completing the assessment

Notes

- No limitations on libraries or frameworks from the chosen language

Data Source

The data source for the API is an Oracle database with a very simple schema and pre-populated with data. The connection details for the database will be provided to each candidate separately. The image below describes the database tables that will be used.



Employee
ID (Number) (PK)
Firstname (Varchar)
Lastname (Varchar)
Badge_Number (Number) (FK)
Country_Code (Varchar)
Job_Title_Code (Number) (FK)
Start_Date (Date)
Leave_Date (Date)

Badge
Badge_Number (Number) (PK)
Badge_Status (Varchar)
Badge_Expiry_Date (Date)

Job_Title
Job_Title_Code (Number) (PK)
Job_Title_Name (Varchar)
Department_Codumber) (FK)

Department
Department_Code (Number) (PK)
Department_Name (Varchar)

API Specification

The API developed should match the following basic specifications. The example JSON object for each entity should be matched in the response of each request. Specific error messages are optional but HTTP status codes described should be adhered to as much as possible. For all requests, 200 should be returned when successful and 401 when username or password are invalid.

Departments

```
{
  department_code: 1
  department_name: 'Example'
}
```

GET /department

- Return all system departments

Badges

```
{
 badge_number: 1,
 badge_status: 'Example',
 badge_expiry_date: 'Date String'
}
```

GET /badges

- Return all system badges

GET /badges/active

- Return only badges whose badge_status is ‘Active’ and whose badge_expiry_date is not in the past
- 404 if no badges are active

GET /badges?badge_number=[badge_number]

- Return the badge matching the badge_number query param
- 404 if no badge_number matches
- 422 if query param is not correct or the passed value is non numeric

Job Titles

```
{
 job_title_code: 1,
 job_title_name: 'Example',
 department_name: 'Example'
}
```

GET /job_titles

- Return all system job_titles

GET /job_titles/:department_name

- Return all job titles from with the department_name URL param
- 404 if no job_titles are in that department
- 422 if the deparment_name passed is not a valid department

Employees

```
{
 id: 1,
 firstname: 'Firstname',
 lastname: 'Lastname',
 badge_number: 1,
 country: 'Country Name',
 job_title: 'Job title name',
 department: 'Department name',
 start_date: 'Date String',
 leave_date: 'Date String or null'
}
```

GET /employees

- Return all system employees

GET /employees/active

- Return all employees who DO NOT have a leave_date set or have a leave_date set in the past
- 404 if there are no active employees

GET /employees?department_name=[department_name]

- Return all employees from a specific department
- 404 if no employees are in that department
- 422 if the department provided is not valid or the query param is incorrect

Notes

- Employee country codes are stored to match an external source. To retrieve country names from that code use the API @ https://restcountries.eu