Organization Directory REST API – ODRA

Minor Project

IBM Career Education

Disclaimer

This Software Requirements Specification document is a guideline. The document details all the high level requirements. The document also describes the broad scope of the project. While developing the solution if the developer has a valid point to add more details being within the scope specified then it can be accommodated after consultation with IBM designated Mentor.

INTRODUCTION

The purpose of this document is to define scope and requirements of a Organization Directory REST API (ODRA) for the developers of a software services company, engaged in developing Java Servlet/JSP based workflow applications. The proposed APIs will provide a loosely coupled mechanism to integrate the existing organization directory framework with the workflow systems being developed.

This document is the primary input to the development team to architect a solution for this project.

System Users

The entire team of developers of the software services company is expected to benefit from the organization directory REST API, ORDA.

Assumptions

1. It is assumed that a module for organization directory management is available. For this project, create a set of tables as outlined in this document and populate them with some meaningful data.
2. To simplify the scope, you may exclude the need to authenticate for API usage.

REQUIREMENTS

ORDA will provide a very simple to use REST APIs for integration with any new workflow system. Learn more about REST at http://en.wikipedia.org/wiki/Representational_state_transfer URL. Each API will return data in JSON format. Learn more about JSON at http://www.json.org/ URL.

The organization directory's database in terms of tables and their respective fields is outlined below:

Table	Description	Fields
Department	Master maintaining departments of an organization. It is assumed that there is no hierarchical structure in the department.	Id Name
Location	Master maintaining locations from where the organization is operating. It is assumed that there is not hierarchical structure in locations.	Id Name Address
Employee	Master maintaining the employee details. It also maintains the reporting relationships. There are 2 kinds of supervisors viz.	Id Name Department Location

Table	Description	Fields
	administrative and functional (can be the same also). Note: In real life, there will be many more fields in this table.	Admin Supervisor (employee id) Functional Supervisor (employee id) e-mail
Location Head	Master maintaining location heads.	Location Id Employee Id
Dept Head	Master maintaining departmental heads.	Department Id Employee Id

The APIs will return the required data in JSON format from the relevant table(s) described above. The list of APIs to be developed is outlined in the following table.

API	Description
GET od/departments	Returns the list of departments.
GET od/locations	Returns the list of locations.
GET od/employees	Returns the list of employees.
GET od/departments/<id>	Returns the details of the department having <id> id.
GET od/locations/<id>	Returns the details of the location having <id> id.
GET od/employee/<id>	Returns the details of the employee having <id> id.

The JSON output structure of each API needs to be developed. A sample structure for “GET od/departments” is outlined below as an guidelines for the rest of the APIs.

```
[
    {
        "generated_at": "2010-07-15T22:31:11Z",
        "departments": [
            {
                "name": "Finance",
                "id": "1"
            },
            {
                "name": "Sales",
                "id": "2"
            },
            {
                "name": "Manufacturing",
                "id": "3"
            },

        ]
    }
]
```

In the event of error, a properly formatted JSON error must be returned.

Testing Guidelines

Each output must be validated using a JSON validation tool like http://jsonlint.com/. In addition, the JSON must be evaluated (using a simple test-bed) to check if each fields' value can be read properly.

DEVELOPMENT ENVIRONMENT

ORDA will be developed as a web application using Java/JSP/Servlets/Filters and DB2 database. Eclipse will be used as the IDE for the same.