Staffing Agency API Documentation

Introduction

The Staffing Agency API offers operations on various entities including Clients, Candidates, Jobs, and Applications. It follows the principles of REST, employs resource-oriented URLs, accepts form-encoded request bodies, returns JSON-encoded responses, and uses standard HTTP protocols.

Base URL:

All API requests are initiated to: http://localhost:8080/

Data Management

The Staffing Agency API interacts with data stored in a database which is managed using MySQL Workbench.

The API performs Create, Read, Update, and Delete (CRUD) operations on the data using SQL queries executed through MySQL Workbench. The operations are designed in a manner that they align with REST principles, offering a straightforward way for clients to interact with the database.

Here's a brief overview of how the API interacts with the database:

Create (POST requests): When a POST request is made, the API executes an INSERT INTO statement to add a new row to the relevant table in the database.

Read (GET requests): GET requests result in a SELECT statement being executed to retrieve data from the database.

Update (PUT requests): For PUT requests, an UPDATE statement is executed to modify existing data in the database.

Delete (DELETE requests): DELETE statements are executed in response to DELETE requests to remove data from the database.

Authentication

Authentication to access and interact with the data via the API is secured with a username and password. These credentials are used to establish a connection with the database via MySQL Workbench.

The credentials for accessing the database are stored in a separate YAML configuration file, which is not exposed in the API. This helps to ensure the security and confidentiality of the access credentials. The username and password are loaded from this configuration file when initiating a session with the database.

Error Handling

In case of an error, the API will return an error message in this format:

Json

```
{
"message": "Description of the error",
"status": HTTP_STATUS_CODE
}
```

Resources

This API facilitates operations on the following resources:

1. Clients

Currently, this entity supports the POST operation and is under active development for full CRUD functionality.

POST /clients: Create a new client.

2. Candidates

The Candidate entity is fully functional with support for all CRUD operations.

GET /candidates: Retrieve a list of all candidates.

GET /candidates/{id}: Retrieve a specific candidate by id.

POST /candidates: Create a new candidate.

PUT /candidates/{id}: Update a specific candidate by id.

DELETE /candidates/{id}: Delete a specific candidate by id.

Candidate Object:

```
{
"candidateId": "Integer",
"name": "String",
"skills": "String",
"applicationIds": ["Integer"]
}
```

3. Jobs

Currently, this entity supports the POST operation and is under active development for full CRUD functionality.

POST /jobs: Create a new job.

4. Applications

Currently, this entity supports the POST operation and is under active development for full CRUD functionality.

POST /applications: Create a new application.

Eventually all entities will be capable of all CRUD operations.

Here are the relationships among the entities:

A Client can post multiple Jobs, but each Job is posted by one Client.

A Candidate can make multiple Applications, but each Application is made by one Candidate.

Each Job can have multiple Applications, but each Application is made for one Job.