E-DBA External Database Interface

This document is to provide interfaces of the external database in section 4.3 of SE_SDW_Project Long Description.

1 Interfaces of an organization

Here lists 6 Interfaces of a organization named "Test-A". The following interface information is provided for student testing purposes only. The instructors will use a different set of interface information for testing. Please do not hardcode the interface information into your code.

Backend API URL: http://172.16.160.88:8001

- `POST /hw/student/authenticate` → Verify name & ID

  input: {"name":"string","id":"string","photo":file}

  output: {"status":"string"}

- `POST /hw/student/record` → Get student GPA & year info

  input: {"name":"string","id":"string"}

  output: {"name":"string","enroll_year":"string", "graduation_year": "string", "gpa": float}

- `POST /hw/thesis/search` → Search thesis by keyword

  input: {"keywords":"string"}

  output: {"title":"string","abstract":"string"}

- `POST /hw/thesis/pdf` → Get thesis PDF path

  input: {"title":"string"}

  output: {file}/{"PDF not found for given title."}

- `POST /hw/bank/authenticate` → Bank account authentication

  input: {

  "bank": "string"

  "account_name": "string",

  "account_number": "string",

"password": "string"

}

output: {"status":"success"}/{"status":"fail"}

- `POST /hw/bank/transfer` → Simulate bank transfer

  input: {

  "from_bank": "string",

  "from_name": "string",

  "from_account": "string",

  "password": "string",

  "to_bank": "string",

  "to_name": "string",

  "to_account": "string",

  "amount": int

  }

  output: {"status":"success"}/{"status":"fail","reason":""}

2 Steps for Implementing `student` and `thesis` Interfaces

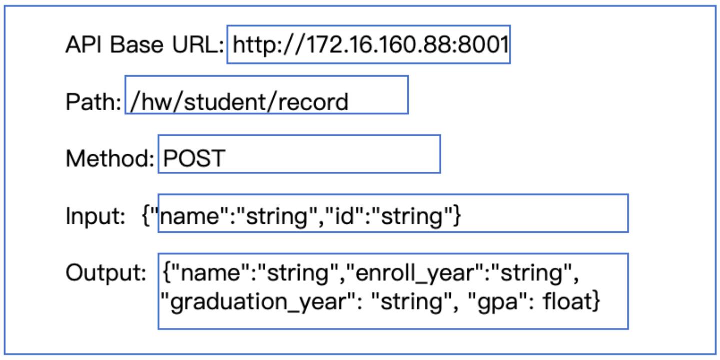
In this document we demostrate the interface `/hw/student/record` as an example.

2.1. Design a Frontend Interface for Data Providers to Fill in Interface Information:：

Provide a user interface that allows registered organizations to fill in the basic information of their interfaces, including:

- API Base URL（如 `http://172.16.160.88:8001`）
- Path（如 `/hw/student/record`）
- Request Method（如 `POST`）
- Request inputs(JSON format)
- Request outputs(JSON format)

The following UI design can be used as a reference:


API Base URL: http://172.16.160.88:8001

Path: /hw/student/record

Method: POST

Input: {"name":"string","id":"string"}

Output: {"name":"string","enroll_year":"string", "graduation_year": "string", "gpa": float}

** Submit this information to the backend and save it to the database.*

2.2. E-DBA System Saves Interface Information to the Database

Store the information filled in by the organization into the database. For example, you can design a table `api_config`:

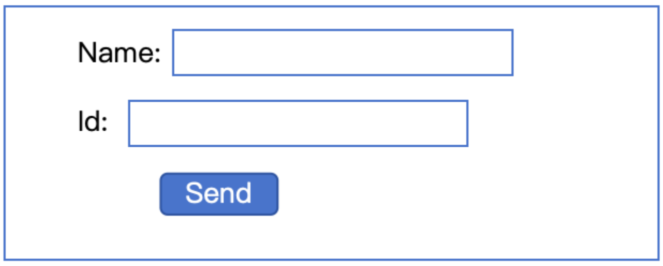
```sql
CREATE TABLE api_config (
    id SERIAL PRIMARY KEY,
    institution_id INT NOT NULL,
    base_url VARCHAR(255) NOT NULL,
    path VARCHAR(255) NOT NULL,
    method VARCHAR(10) NOT NULL,
    input JSONB,
    output JSONB,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

2.3. E-DBA System Backend Dynamically Calls Interfaces

Create a service layer that queries the interface parameters from the database and returns the query results.

2.4. Provide a Frontend UI to Render and Provide Configured Query Services

- The UI renders based on the request input fields. Since the input in the interface information consists of two fields, the UI displays input boxes for `name` and `id`. Provide an input interface for data consumers, as shown below (For other interfaces, you must have different UI to get data, if data provider config 3 inputs for this interface, you should display 3 input areas in your UI):



- The backend creates actual query parameters based on user input and calls the interface provided by the organization, then returns the results.
- For example, when a user inputs a query for a student via the frontend with the following parameters:

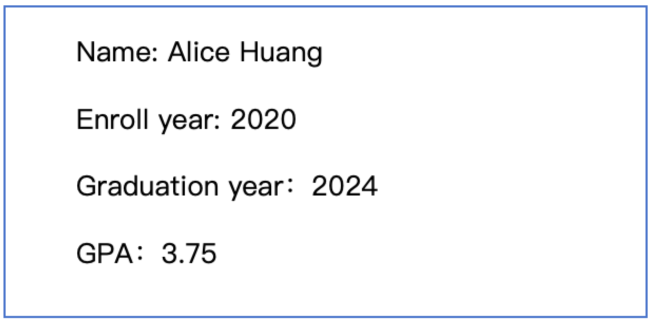
```
{
    "institution_id": 1,
    "name": "Alice Huang",
    "id": "S20230001"
}
```

- The backend looks up the corresponding interface configuration based on `institution_id` and calls the interface of organization "Test-A". The following code is for demo purposes only and needs to be further refined by students. Note: If data provider from another organization provide more or less inputs, you need to

```

base_url = 'http://10.0.7.11:8001' # should be got from database
path = '/hw/student/record'  # should be got from database
method = 'Post'
url = base_url + path

inputs = {"name":"string","id":"string"}
keys_list = list(inputs.keys()) # should be got from database

data_to_send = { # should be got from your UI
    "name": "Alice Huang",
    "id": "S20230001"
}

dynamic_data = {key: f"example_value_for_{key}" if key not in data_to_send else data_to_send[key] for key in inputs.keys()}
```

```python
final_payload = {key: data_to_send.get(key, "") for key in inputs.keys()}

# in this example, it will print:  {'name': 'Alice Huang', 'id': 'S20230001', 'age': 12}
print(final_payload)

if method.lower() == 'post':
    response = requests.post(url, json=final_payload)
elif method.lower() == 'get':
    response = requests.get(url, params={"example_param": "value"})
else:
    print("Unsupported HTTP method:", method)
    response = None
```

- The backend would get a result `response` from the interface we given and you can display in your UI. In this example, you may see following Info in your frontend UI:



Name: Alice Huang

Enroll year: 2020

Graduation year：2024

GPA：3.75

2.5. Data Provider Configuration Change

If a private data provider needs to change the interface configuration, they can simply repeat the steps outlined in section 2.1. Your system will not be affected, and there is no need to stop the running of E-DBA. All configurations should be loaded dynamically.

3 Bank Interface Implementation Steps

- Write the bank interface information to a text document, and the E-DBA system directly reads this information to perform bank-related operations. （Text file will be provided on iSpace)
- When you want to change the bank information, simply modify the text information.

4 Fail to configure

If students are unable to dynamically configure the interfaces, they can hardcode the interface information into the code, but points will be deducted.

```python
requests.post("http://10.0.7.11:8001/hw/student/record", json={
    "name": "Alice Huang",
    "id": "S20230001"
})
```

Appendix： Part of data in External database

Following is part of data in external database for testing purpose. You can try to access to the interfaces and look for the data you want.

You can also access to the webpage: http://172.16.160.88:8899/tester.html to get the following test data.

Student data:

Note: test photos are provied on iSpace, you can download

```json

[
  {
    "name": "Alice Huang",
    "id": "S20230001",
    "enroll": "2020",
    "grad": "2024",
    "gpa": 3.75,
    "photo": "static/photos/alice.jpg"
  },
  {
    "name": "Brian Chen",
    "id": "S20230002",
    "enroll": "2019",
    "grad": "2023",
    "gpa": 3.6,
    "photo": "static/photos/brian.jpg"
  },
  {
    "name": "Cindy Lin",
    "id": "S20230003",
    "enroll": "2021",
    "grad": "2025",
    "gpa": 3.9,
    "photo": "static/photos/cindy.jpg"
  },
]
```

Thesis data:

```

[
  {
    "title": "AI in Education and Learning Systems",
    "abstract": "This paper discusses the topic: AI in Education and Learning Systems in detail."
  },
  {
    "title": "Blockchain Applications in University Records",
    "abstract": "This paper discusses the topic: Blockchain Applications in University Records in detail."
  },
  {
    "title": "IoT Security in Smart Campus",
    "abstract": "This paper discusses the topic: IoT Security in Smart Campus in detail."
  },
  {
    "title": "Green Energy Solutions for Schools",
    "abstract": "This paper discusses the topic: Green Energy Solutions for Schools in detail."
  },
  {
    "title": "Edge Computing for Classroom Analytics",
    "abstract": "This paper discusses the topic: Edge Computing for Classroom Analytics in detail."
  },
  {
    "title": "Cloud-based Student Portfolios",
    "abstract": "This paper discusses the topic: Cloud-based Student Portfolios in detail."
  },
]
```

Bank data:

```
[
  {
    "name": "Campus Cash Cooperative",
    "account": "641387141765274",
    "password": "8799",
    "bank": "Global Education Bank",
    "balance": 26932724.07
  },
  {
    "name": "Utopia Credit Union",
    "account": "670547811218584",
    "password": "9978",
    "bank": "FutureLearn Federal Bank",
    "balance": 29619364.04
  },
  {
    "name": "EdGrow Finance Co.",
    "account": "393077718917153",
    "password": "9217",
    "bank": "Bank of Utopia",
    "balance": 38575793.4
  },
  {
    "name": "EdGrow Finance Co.",
    "account": "137070703603485",
    "password": "1851",
    "bank": "Global Education Bank",
    "balance": 16585806.35
  },
  {
    "name": "BrightMind Capital",
    "account": "265254690447221",
    "password": "4664",
    "bank": "Global Education Bank",
    "balance": 14797714.87
  },
  {
    "name": "Scholars Advantage Trust",
    "account": "904622106460646",
    "password": "1962",
    "bank": "Continental Scholars Bank",
    "balance": 45367154.94
  },
]
```