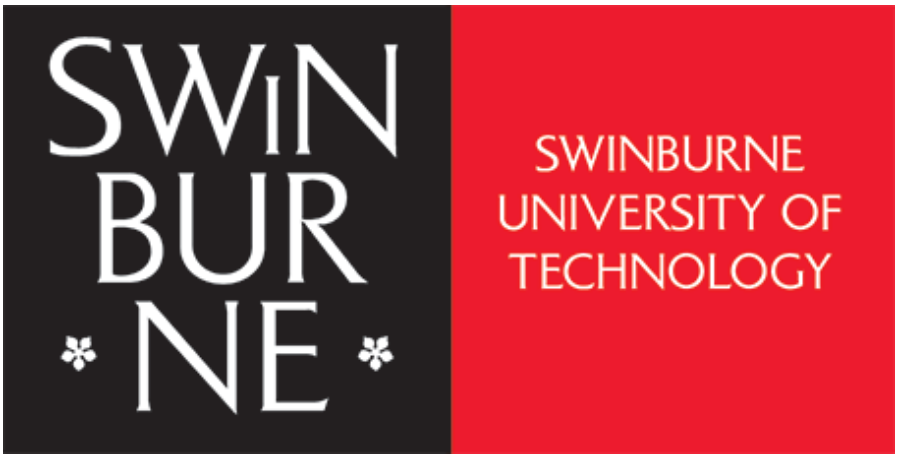


2023-COS30049-Computing Technology Innovation Project

Workshop Guide

Workshop 06

Introduction to Relational Databases and Fetching Backend Data from the Frontend

Objective: By the end of this workshop, students should have an understanding of the basic concepts of MySQL Database and be able to connect database connections on FastAPI. In addition, SQL query commands will be introduced in this week's workshop material in order to let students retrieve data from the database.

Workshop Structure:

1. MySQL Introduction and Configuration（20 mins）:

   MySQL is an open-source relational database management system (RDBMS) widely used for building and managing various types of applications and websites. It offers key features and advantages, including open-source nature, relational database structure, cross-platform support, high performance, scalability, multi-language support, security, and an active community. MySQL is commonly used in web development, enterprise applications, embedded systems, cloud computing, and big data applications. It adheres to the ACID properties, ensuring data integrity and consistency.

   Download the MySQL on Mac:

https://dev.mysql.com/downloads/mysql/
Config Your MySQL on Mac:
https://dev.mysql.com/doc/refman/8.0/en/macos-installation.html

Download the MySQL on Windows:
https://dev.mysql.com/downloads/mysql/
Config Your MySQL on Windows:
https://dev.mysql.com/doc/refman/8.0/en/windows-installation.html

After the installation, please note that you need to remember your account name (root) and password for future database operations.

2. MySQL connection (40 mins)

- STEP 1: Connect the MySQL in the terminal:

  mysql -u <username> -p

  or

  /usr/local/mysql/bin/mysql -u root -p（for Mac）

  common issue: windows command not found: mysql

  https://phoenixnap.com/kb/mysql-command-not-found-error

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

- STEP 2: Create a database

  CREATE DATABASE database_name;

  - CREATE DATABASE is the command to create a database.
  - database_name is the name you want to give to your new database.

```
mysql> CREATE DATABASE mydb;
Query OK, 1 row affected (0.00 sec)
```

- STEP 3: Show existed databases

  To list all the databases in MySQL, you can use the

  SHOW DATABASES;

```
mysql> CREATE DATABASE mydb;
Query OK, 1 row affected (0.00 sec)

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mydb               |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.01 sec)
```

- STEP 4: Select existed database

  To select a specific database in MySQL, you can use the *USE* statement.

```
mysql> USE mydb
Database changed
```

3. Data Manipulations in MySQL (20 mins)

In MySQL, there are many data types that refer to different attributes that users defined in the table. Here are some common types in MySQL

- INT or INTEGER: Integer type for whole numbers.
- FLOAT: Floating-point type for approximate numeric values.
- DOUBLE: Double-precision floating-point type.
- VARCHAR(size): Variable-length character string.
- DATETIME: Date and time in 'YYYY-MM-DD HH:MM:SS' format.
- DATE: Date in 'YYYY-MM-DD' format.
- BOOLEAN or BOOL: Boolean type that can hold values TRUE, FALSE, and NULL.

1) Create a table:

According to these common types above, we can create a student table to store student-related data.

Using *CREATE TABLE students* command to create a table

```
mysql> CREATE TABLE students (
    ->   student_id INT AUTO_INCREMENT PRIMARY KEY,
    ->   first_name VARCHAR(50) NOT NULL,
    ->   last_name VARCHAR(50) NOT NULL,
    ->   date_of_birth DATE,
    ->   phone_number VARCHAR(15)
    -> );
Query OK, 0 rows affected (0.01 sec)
```

2) Insert data into the table

According to the data structure above, we can insert the data into the database with the command:

INSERT INTO students

```
mysql> INSERT INTO students (first_name, last_name, date_of_birth, phone_number)
    -> VALUES ('John', 'Doe', '2000-01-15', '123-456-7890');
Query OK, 1 row affected (0.00 sec)
```

3) List all data inside the students table

*SELECT * FROM students;*

```
mysql> SELECT * FROM students;
+------------+------------+-----------+---------------+--------------+
| student_id | first_name | last_name | date_of_birth | phone_number |
+------------+------------+-----------+---------------+--------------+
|          1 | John       | Doe       | 2000-01-15    | 123-456-7890 |
+------------+------------+-----------+---------------+--------------+
1 row in set (0.00 sec)
```

4) List specific columns in the students table

SELECT first_name, last_name FROM students;

```
mysql> SELECT first_name, last_name FROM students;
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| John       | Doe       |
+------------+-----------+
1 row in set (0.00 sec)
```

5) Modify the data inside the students table

UPDATE students

SET first_name = 'Jack'

WHERE student_id = 1;

```
mysql> SELECT first_name, last_name FROM students;
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| John       | Doe       |
+------------+-----------+
1 row in set (0.00 sec)

mysql> UPDATE students
    -> SET first_name = 'Jack'
    -> WHERE student_id = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM students;
+------------+------------+-----------+---------------+--------------+
| student_id | first_name | last_name | date_of_birth | phone_number |
+------------+------------+-----------+---------------+--------------+
|          1 | Jack       | Doe       | 2000-01-15    | 123-456-7890 |
+------------+------------+-----------+---------------+--------------+
1 row in set (0.00 sec)
```

6) Delete the data inside the students table

DELETE FROM students

WHERE first_name = 'Jack';

```
mysql> DELETE FROM students
    -> WHERE first_name = 'Jack';
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM students;
Empty set (0.00 sec)
```

4. Config MySQL in the FastAPI (20 mins)

In FastAPI, when we need to connect the MySQL database, we need to config the database information in the file first.

```
# MySQL database connection configuration
db_config = {
    "host": "localhost",
    "user": "your username",
    "password": "your password",
    "database": "mydb"
}
```

In the database configuration, we need to define a JSON object that includes:

- host: the host address of the database
- user: username to login the MySQL
- password: the corresponding password for the user
- database: the name of database that you want to connect

After we config the database properly, we need to install the dependency to connect MySQL

conda install -c anaconda mysql-connector-python

Then, we need to import this package in the top of the file

```
# DB
import mysql.connector
```

After we config and import the related dependency, we can write an interface to test if the data can be retrieved.
Here are the steps for you to connect the MySQL, you will need to follow the steps:

Step1 :

```
# Establish a database connection
connection = mysql.connector.connect(**db_config)
```

Step2 :

```
# Create a cursor to execute SQL queries
cursor = connection.cursor()
```

Step3:

```
# Define the SQL query to retrieve data (e.g., all
students)
query = "SELECT * FROM students"
```

Step4 :

```
# Execute the SQL query
cursor.execute(query)
```

Step5 :

```
# Fetch all the rows
result = cursor.fetchall()
```

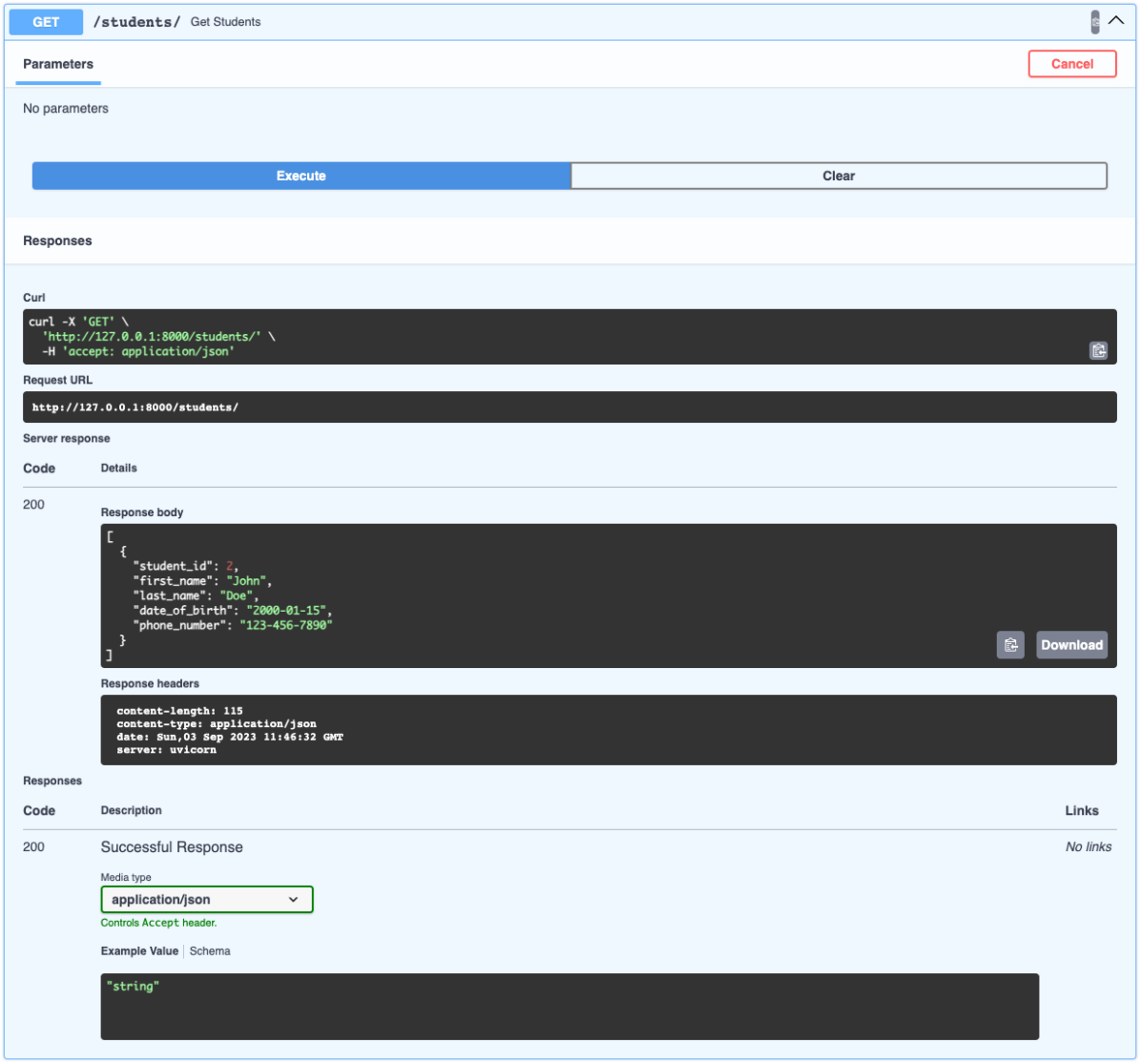
Step6 :

```
# Convert the result to a list of dictionaries
students = [dict(zip(cursor.column_names, row)) for
row in result]
```

Step7 :

```
# Close the cursor and the database connection
cursor.close()
connection.close()
```

With the above given steps, you can write an interface to get all students' data. And when you check this interface in FastAPI documents, this should be like this:



Code sample can be found in: https://codesandbox.io/p/sandbox/fastapi-demo-kkd6yv?file=/week6/week6.py:17,24

5. Connection Between Front-end and Back-end Components (20 mins)

What is Axios ?

Axios is a Promise-based HTTP client used for making HTTP requests in both browser and Node.js environments. It's a popular JavaScript library that allows developers to interact

with servers for data exchange, offering features like a simple API, Promise support for asynchronous operations, automatic data conversion, interceptors, request cancellation, and robust error handling. Axios is widely adopted in both frontend and backend development for communicating with backend APIs, fetching data, and sending data.

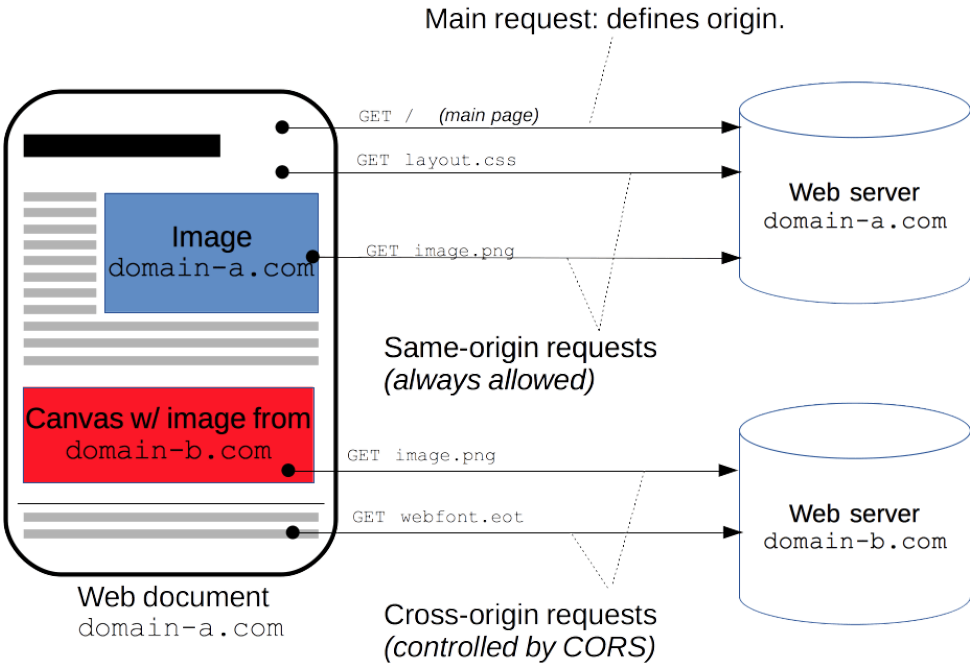
- Security: Axios supports secure communication through features like automatic handling of cookies and cross-site request forgery (CSRF) protection, making it a reliable choice for handling authentication and sensitive data transmission.
- Timeouts: You can set timeouts for Axios requests to ensure that requests do not hang indefinitely, providing control over request responsiveness.
- Browser Compatibility: Axios is designed to work seamlessly in modern web browsers, handling cross-origin requests and adhering to browser security policies.

Install the AXIOS package in the React Folder:
npm install axios

What is CORS ?

Cross-Origin Resource Sharing (CORS) is a web browser security mechanism used to control whether one web page can access specific resources on another domain. This security feature is a result of the Same-Origin Policy, which restricts web pages from accessing resources from different

origins. CORS allows servers to include special HTTP headers in responses to indicate which origins are permitted to access their resources. By configuring CORS policies, web developers enhance security while enabling legitimate cross-origin communication. It involves concepts like the Same-Origin Policy, CORS headers, preflight requests, and contributes to a safer web environment.



Here is the official documentation: https://fastapi.tiangolo.com/tutorial/cors/?h=%20cors#use-corsmiddleware

In the FastAPI project, you will need to add the following lines:

```
from fastapi.middleware.cors import CORSMiddleware


app = FastAPI()


origins = ["*"]


app.add_middleware(
      CORSMiddleware,
```

```
        allow_origins=origins,
        allow_credentials=True,
        allow_methods=["*"],
        allow_headers=["*"],
    )
```

Here is the code example: https://codesandbox.io/p/sandbox/fastapi-demo-kkd6yv?file=/week6/week6.py

In the React project, you will need to add the following lines to request the API in GET method:

```
    handleButtonClick = () => {
        axios
        .get("http://127.0.0.1:8000/jsonData")
        .then((response) => {
            console.log(response.data);
            })
        .catch((error) => {
            console.error("there are errors:", error);
        });
    };
```

Here is the code example: https://codesandbox.io/p/sandbox/fastapi-demo-kkd6yv?file=/week6/app.js:6,3-15,5

6. Reflection (5 mins)

Give students the opportunity to share what they learned, found interesting, or had difficulty understanding. Offer additional resources for them to learn more about FastAPI Python Framework

Online Code Sample:
https://codesandbox.io/p/sandbox/fastapi-demo-kkd6yv?file=/week6/week6.py:17,24

MySQL Connector/Python Developer Guide
https://dev.mysql.com/doc/connectors/en/connector-python.html

Common commands in Anaconda
https://www.python-engineer.com/posts/anaconda-basics/

CORS (Cross-Origin Resource Sharing)
https://fastapi.tiangolo.com/tutorial/cors/?h=%20cors#use-corsmiddleware

Installing MySQL on Microsoft Windows
https://dev.mysql.com/doc/refman/8.0/en/windows-installation.html

MySQL common Queries
https://dev.mysql.com/doc/refman/8.0/en/tutorial.html