Lab 05. Update and Delete Data

Objectives:

- Learn how to implement routes for updating and deleting data
- Learn how to implement APIs to update and delete data from MongoDB

A. Create a New Project

1. Create a folder `lab05` with a new python file `main.py` in it.
2. Implement a home route in `main.py`.
3. Execute `main.py` file to run the web server.

B. Routes for Update, Delete

It is common for APIs to use HTTP methods `UPDATE` and `DELETE` to for routes to update and delete data respectively. Let's implement these 2 routes without connecting to MongoDB.

1. We will create a model class `Book`, which will help us with validation.

```
class Book(BaseModel):
    title: str
    authors: List[str] = None
    tags: List[str] = None
    pages: Optional[int] = None
    publishes: Optional[int] = None
```

2. Create a `PUT /books/{book_id}` route which receives a `book` instance from request body, and a string `book_id` from path.
    - The `book_id` gives the ID of the book to be updated.
    - The `book` gives values to replace the original data.
    - Implement the route to return a JSON object combining `book` and `book_id`.

```
@app.put('/books/{book_id}')
def update_book(book_id: str, book: Book):
  d = book.dict()
  d['_id'] = book_id
  return d
```

3. Create a `DELETE /books/{book_id}` route which receives a string `book_id`.
    - The `book_id` gives the ID of the book to be deleted.
    - Implement the route to return the `book_id`.

```
@app.delete('/books/{book_id}')
def delete_book(book_id: str):
  return {'_id': book_id}
```

4. Test your routes using http://localhost:8000/docs.

C. Connect to MongoDB

1. Create a file `database.py` with following code
    - Use your own connection string with correct username, password and database

```
from pymongo import MongoClient

# MongoDB attributes
mongodb_url =
'mongodb+srv://root:qwer1234@cluster0.hlixs.mongodb.net__/demo?
retryWrites=true&w=majority'

try:
  client = MongoClient(mongodb_url)
  db = client['demo']
  print('Connected to MongoDB')
except Exception as e:
  print(repr(e))
  raise Exception('Error in MongoDB connection.')
```

2. Import `database.py` file in `main.py`.

```
from database import *
```

Update a Book

1. Modify the route `PUT /books/{book_id}` to update data in MongoDB.
    - A model can be converted to a dictionary using its `dict()` method. More info at https://pydantic-docs.helpmanual.io/usage/exporting_models/
    - The API returns the ID of inserted document. Remember to convert it from ObjectId to string using `str()`.

```
@app.put('/books/{book_id}')
def update_book(book_id: str, book: Book):
  try:
    result = db.books.replace_one({'_id':ObjectId(book_id)}, book.dict())

    return {'matched_count': result.matched_count, 'modified_count':
result.modified_count}
  except Exception as e:
    print(repr(e))
    return JSONResponse({'error': str(e)}, 500)
```

Delete a Book

2. Modify the route `DELETE /books/{book_id}` to delete an item in MongoDB
    - The `book_id` is used to identify an book.
    - It returns an integer indicating number of books deleted.

```python
@app.delete('/books/{book_id}')
def delete_book(book_id: str):
  result = db.books.delete_one({'_id': ObjectId(book_id)})
  return {'deleted_count': result.deleted_count}
```

Reference

- https://api.mongodb.com/python/current/api/pymongo/collection.html