NAME :- DNYANG BAGLA
PANEL :- C PC-33
C-3

FINAL YEAR BTECH.

PAGE NO.

BDA LAB ASSIGNMENT - 3

AIM :- Perform database connectivity with mongodb as backend and any frontend from PHP/Python/Java.

OBJECTIVE :-

- To learn connectivity of mongodb as backend and Python/Java as frontend.
- To execute CRUD operations.

THEORY :-

(1) Introduction to Pymongo :-

Pymongo is a python distribution which provides tools to work with mongodb., it is most preferred way to connect with mongodb database from python.

(2) Establishing connection :-

We use mongoclient instance and specific database name. If database doesn't exist it will create it and connect it. → from pymongo import MongoClient.

client = MongoClient("localhost", 27017).

The above code will connect to default host & port.

(3) Accessing database :-

db = client['datamap']

in pymongo dictionaries are used to represent documents retrieving

(4) Retrieving documents :-

findone returns a single document matching query as one if it doesn't exists, it returns first match for finding all documents in collection using the find method.

(5) Updating document :-
documents can be updated using update-one() first parameter taken by this function is a query object defining document to be updated. If there are more than one then only the first one is updated.
Inserting

(6) document :-
We use insert_one() method when document is inserted a special key-id is generated and is unique to this document.

(7) Deleting document :-
delete_one() is used to delete one document in mongodb the first parameter for this method is query object.

(8) Dropping a collection :-
db.collection-name.drop()

* INPUT :- Python program to do CRUD operations for suitable application.
* OUTPUT :- Successful execution of CRUD operation using python as frontend and pymongo as backend.

FAQ'S :-

Q.1 What is pymongo?
=> Pymongo is a python distribution containing tools for working with MongoDB and is recommended way to work with mongo DB.

PAGE NO.:

Q.2 What is mongo engine?

→ MongoEngine is a python library that acts a object document mapped with mongoDB, a NoSQL database. It uses simple declarative API similar to django ORM.

Q.3 What is the purpose of following statement?

→ from pymongo import MongoClient.

To establish a connection to mongoDB with pymongo we use the MongoClient class. PyMongo is the inbuilt ~~function~~ python package through which one can import mongo client.

NAME:- Divyang Bagla

PANEL:- C

ROLL NO.:- PC33

SUBJECT:- BDA

LAB ASSIGNMENT - 3

In [2]:

```
from pymongo import MongoClient
```

In [3]:

```
client = MongoClient('localhost',27017)

# create database
db = client.post_data
```

In [4]:

```
# create collections and insert documnets
posts = db.posts

post_1 = {
    'title': 'Interstellar',
    'genre': 'Sci-fi',
    'director': 'Christopher Nolan'
}
post_2 = {
    'title': 'Reservoir Dogs',
    'content': 'Drama',
    'director': 'Quentin Tarantino'
}

new_result = posts.insert_many([post_1,post_2])
print('Multiple posts: {0}'.format(new_result.inserted_ids))
```

```
Multiple posts: [ObjectId('615802577eb95ee4480e904d'), ObjectId('615802577eb
95ee4480e904e')]
```

In [5]:

```
posts.count_documents({})
```

Out[5]:

```
2
```

In [14]:

```
#find one
bills_posts = posts.find_one({'title' : 'Interstellar'})
bills_posts
```

Out[14]:

```
{'_id': ObjectId('61574ed7a8512f71979a04e3'),
 'title': 'Interstellar',
 'genre': 'Sci-fi',
 'director': 'Christopher Nolan'}
```

In [6]:

```
#update
posts.update_one({'title':'Reservoir Dogs'}, {"$set": {'test_value': 1000}})
```

Out[6]:

```
<pymongo.results.UpdateResult at 0x2a35540ea40>
```

In [8]:

```
#find all
cursor = posts.find()
```

In [9]:

```
for i in cursor:
    print(i)
```

```
{'_id': ObjectId('615802577eb95ee4480e904d'), 'title': 'Interstellar', 'genr
e': 'Sci-fi', 'director': 'Christopher Nolan'}
{'_id': ObjectId('615802577eb95ee4480e904e'), 'title': 'Reservoir Dogs', 'co
ntent': 'Drama', 'director': 'Quentin Tarantino', 'test_value': 1000}
```

In [11]:

```
newMovie = [
    { "title": "Blade Runner 2049", "genre": "Sci-fi, Action", "director": "Denis Villeneuv
    {"title": "Fight Club", "genre": "Action, Drama", "director": "David Fincher"}
]

posts.insert_many(newMovie)
```

Out[11]:

```
<pymongo.results.InsertManyResult at 0x2a355416cc0>
```

In [12]:

```
for i in posts.find():
    print(i)
```

```
{'_id': ObjectId('615802577eb95ee4480e904d'), 'title': 'Interstellar', 'genr
e': 'Sci-fi', 'director': 'Christopher Nolan'}
{'_id': ObjectId('615802577eb95ee4480e904e'), 'title': 'Reservoir Dogs', 'co
ntent': 'Drama', 'director': 'Quentin Tarantino', 'test_value': 1000}
{'_id': ObjectId('615804287eb95ee4480e904f'), 'title': 'Blade Runner 2049',
'genre': 'Sci-fi, Action', 'director': 'Denis Villeneuve'}
{'_id': ObjectId('615804287eb95ee4480e9050'), 'title': 'Fight Club', 'genr
e': 'Action, Drama', 'director': 'David Fincher'}
```

In [13]:

```
# delete
posts.delete_one({'title' : 'Blade Runner 2049' })
```

Out[13]:

```
<pymongo.results.DeleteResult at 0x2a3553f96c0>
```

In [14]:

```
for i in posts.find():
    print(i)
```

```
{'_id': ObjectId('615802577eb95ee4480e904d'), 'title': 'Interstellar', 'genr
e': 'Sci-fi', 'director': 'Christopher Nolan'}
{'_id': ObjectId('615802577eb95ee4480e904e'), 'title': 'Reservoir Dogs', 'co
ntent': 'Drama', 'director': 'Quentin Tarantino', 'test_value': 1000}
{'_id': ObjectId('615804287eb95ee4480e9050'), 'title': 'Fight Club', 'genr
e': 'Action, Drama', 'director': 'David Fincher'}
```

In [18]:

```
#sort output
for i in posts.find().sort("genre"):
    print(i)
```

```
{'_id': ObjectId('615802577eb95ee4480e904e'), 'title': 'Reservoir Dogs', 'co
ntent': 'Drama', 'director': 'Quentin Tarantino', 'test_value': 1000}
{'_id': ObjectId('615804287eb95ee4480e9050'), 'title': 'Fight Club', 'genr
e': 'Action, Drama', 'director': 'David Fincher'}
{'_id': ObjectId('615802577eb95ee4480e904d'), 'title': 'Interstellar', 'genr
e': 'Sci-fi', 'director': 'Christopher Nolan'}
```

In [ ]: