PyMongo: Python and MongoDB Interaction
Briefing Document: PyMongo Introduction
Date: October 26, 2023**Source:** Excerpts from "08 - PyMongo.pdf" by Mark Fontenot, PhD, Northeastern University
Overview:
This document provides a brief overview of PyMongo, a Python library used for interacting with MongoDB databases. The source material introduces the fundamental concepts of using PyMongo, including connecting to a MongoDB instance, accessing databases and collections, inserting documents, and querying data. It also provides instructions for setting up a development environment using JupyterLab.
Main Themes and Important Ideas/Facts:

- **PyMongo as a Python Driver:** The primary purpose of PyMongo is to serve as an interface between Python applications and MongoDB databases. As stated, "*PyMongo is a Python library for interfacing with MongoDB instances*". This allows Python developers to leverage the features of MongoDB within their applications.
- **Establishing a Connection:** The document demonstrates how to establish a connection to a MongoDB instance using the `MongoClient` class from the `pymongo` library. The connection string typically includes the MongoDB server address, port (default is 27017), and optionally, authentication credentials:

```
from pymongo import MongoClient
client = MongoClient(
‘mongodb://user_name:pw@localhost:27017’
)
```

- **Accessing Databases and Collections:** Once a client is established, the document illustrates how to access specific databases and collections. Databases are accessed using dictionary-style or attribute-style access on the `client` object, and similarly, collections are accessed on the `db` object:

```
db = client[‘ds4300’] # or client.ds4300
collection = db[‘myCollection’] #or db.myCollection
```

- **Inserting Documents:** The document provides an example of inserting a single document into a collection using the `insert_one()` method. The document is represented as a Python dictionary:

```
post = { “author”: “Mark”, “text”: “MongoDB is Cool!”, “tags”: [“mongodb”, “python”] }
post_id = collection.insert_one(post).inserted_id
print(post_id)
```

The `insert_one()` method returns an `InsertOneResult` object, and its `inserted_id` attribute contains the unique ID assigned to the newly inserted document.

- **Querying Data (Finding Documents):** The document demonstrates how to retrieve documents from a collection using the `find()` method. It provides an example of finding all movies with the "year" field equal to 2000:

```
movies_2000 = db.movies.find({"year": 2000})
```

The results of the `find()` method are typically a cursor, which can be iterated over. The document also shows how to use `bson.json_util.dumps` to pretty-print the results in JSON format:

```
from bson.json_util import dumps
# Print results
print(dumps(movies_2000, indent = 2))
```

- **Setting up a Development Environment with JupyterLab:** The source emphasizes the use of JupyterLab for practical exercises and provides step-by-step instructions for setting up the necessary environment:
  - Activate a Python environment (conda or venv).
  - Install PyMongo using `pip install pymongo`.
  - Install JupyterLab using `pip install jupyterlab`.
  - Download and unzip a provided zip file containing Jupyter Notebooks.
  - Navigate to the unzipped folder in the terminal.
  - Run JupyterLab using the command `jupyter lab`.

Key Takeaways:

- PyMongo is the official Python driver for MongoDB, enabling seamless interaction between Python applications and MongoDB databases.
- Basic operations include connecting to a MongoDB server, selecting databases and collections, inserting data as Python dictionaries, and querying data using dictionary-based queries.
- The document highlights the importance of setting up a proper development environment, specifically recommending the use of JupyterLab for hands-on practice with PyMongo.

This briefing document provides a foundational understanding of PyMongo based on the provided source material. Further exploration of PyMongo's capabilities would involve examining more advanced querying techniques, data manipulation methods, and error handling.