



---

**FINAL PROJECT REPPORT – NoSQL**  
**Project – CINE STORE**



**Professor: Fatah Ioualitene**

Submitting by  
Yao Junior

Paris, 11 July 2020

## Contents

Background .....	3
The Application .....	4
Data Flow .....	4
Creating Information.....	5
Reading Information .....	6
Updating Information .....	7
Deleting Information.....	8
MongoDB .....	10
The Python Coding.....	11
Conclusion.....	12
Reference .....	12

## Background

A NoSQL (originally referring to "non-SQL" or "non-relational") database provides a mechanism for storage and retrieval of data that is modelled in means other than the tabular relations used in relational databases. MongoDB Inc. (formerly 10gen) is an American software company that develops and provides commercial support for the open source database MongoDB, a NoSQL database that stores data in JSON-like documents with flexible schemas.

Thanks to EPITA, we have chance to explore the software MongoDB through the subject of NoSQL. Cine Store is the idea to develop a simple application which helps to manage the film information database by using MongoDB as the back end.



## The Application

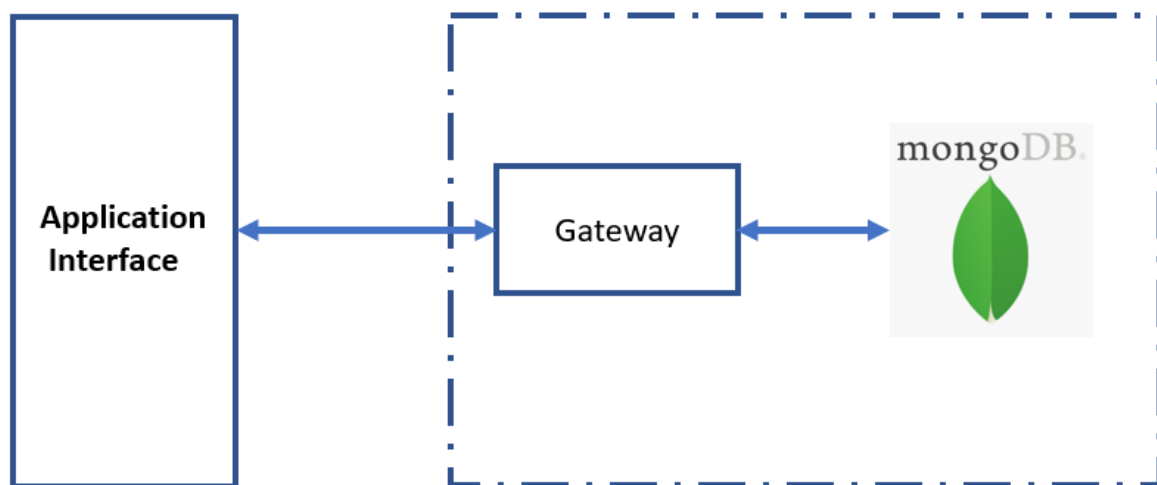
### Data Flow

The application allows us to store any movies information into the MongoDB database. The application is structure into different section the add movie section is used to take the movie details and creation a collection into the database, to update any information from the section delete and update acts as interface

The deletion of any information from the database is based on the name of the movie and the language are the passing value to delete a movie information.

Updating or changing any information value uses the name of the movie as a key, and the description and release data of the movie can be change or update

The application can also read data from the mongo DB and display it into the list all section.



## Creating Information

Insertion of new Movies information

### Insertion SCHEMA

NAME: "take the name of the movie as input"

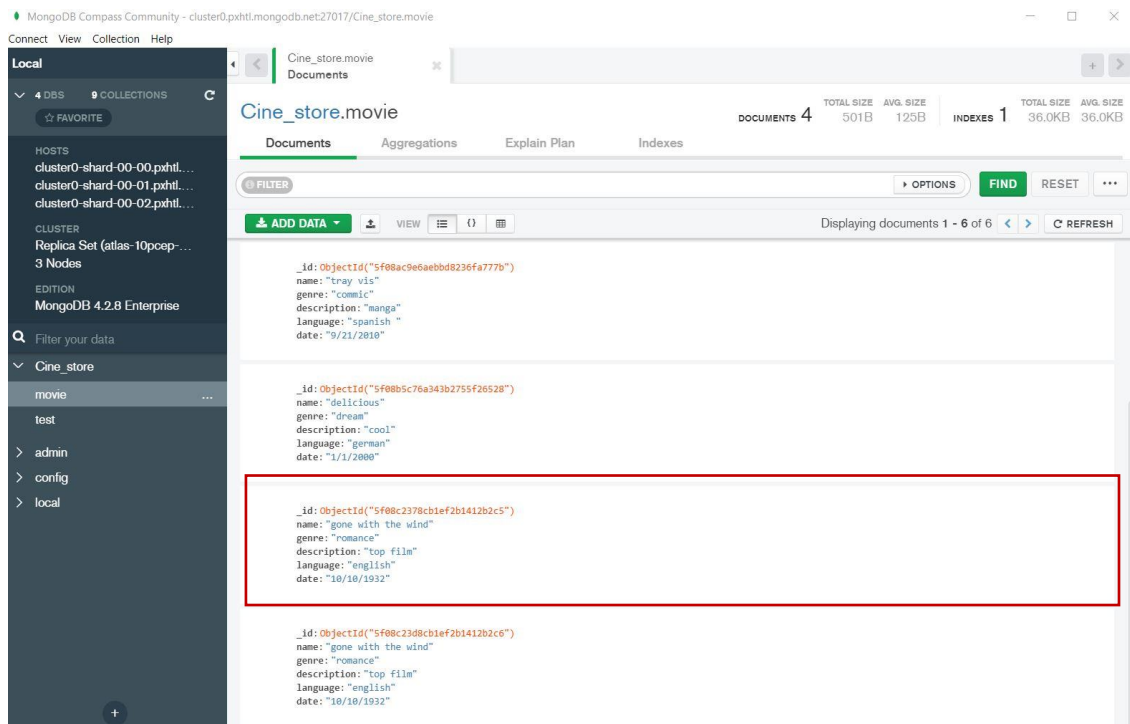
GENRE: "take the type of movie (action)"

DESCRIPTION: "any description of the movie"

LANGUAGE: "insert movie language"

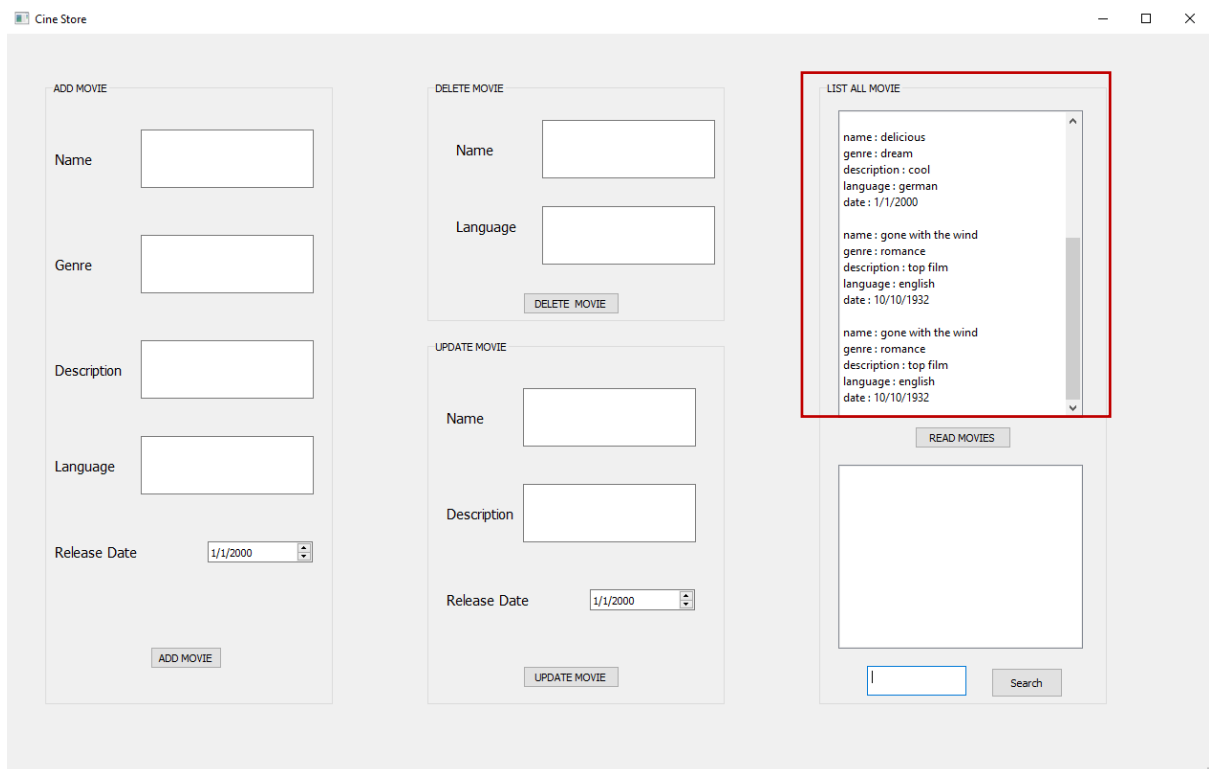
DATE:" insert the release date of the movie"

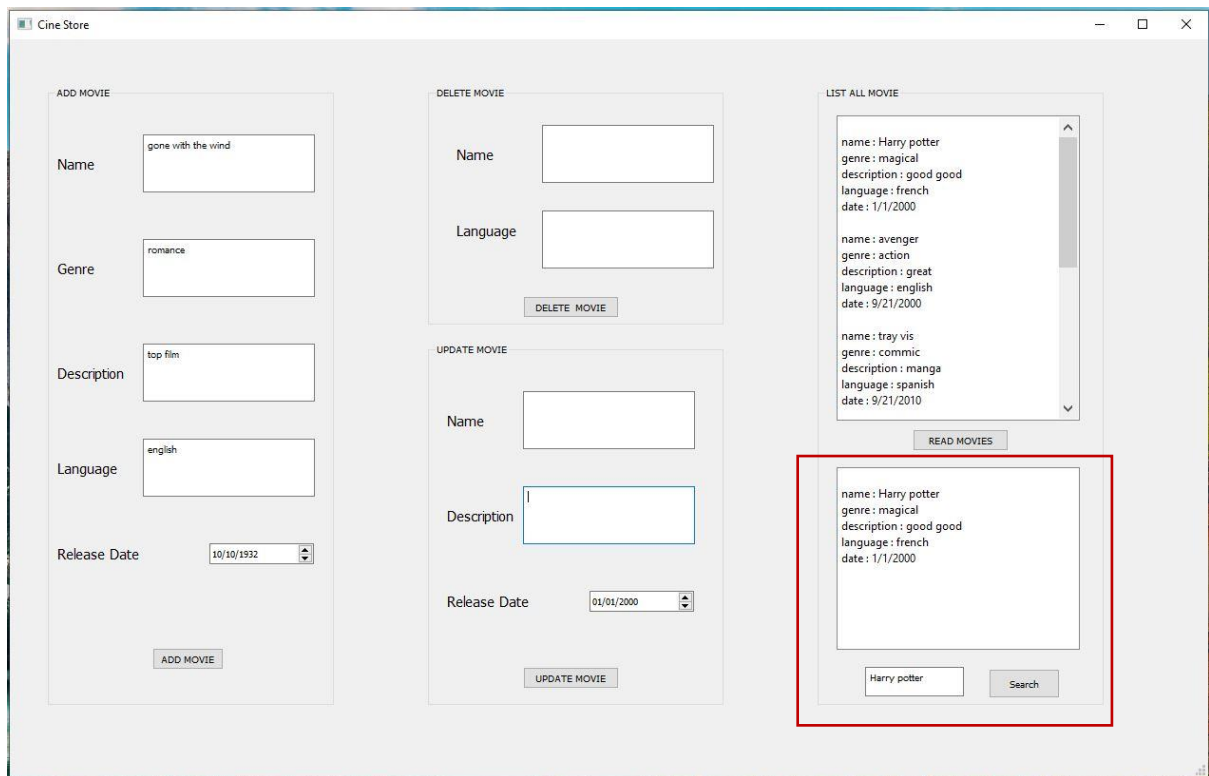
The screenshot displays the 'Cine Store' application window, which is divided into three main functional panels. The 'ADD MOVIE' panel on the left is highlighted with a red border and contains input fields for Name (filled with 'gone with the wind'), Genre (filled with 'romance'), Description (filled with 'top film'), Language (filled with 'english'), and Release Date (a date picker set to '10/10/1932'). An 'ADD MOVIE' button is at the bottom. The 'DELETE MOVIE' panel in the center has input fields for Name and Language, with a 'DELETE MOVIE' button below. The 'UPDATE MOVIE' panel below it has input fields for Name, Description, and Release Date (a date picker set to '01/01/2000'), with an 'UPDATE MOVIE' button at the bottom. The 'LIST ALL MOVIE' panel on the right features a scrollable list of movie details, including 'Harry potter' and 'avenger', each with its genre, description, language, and date. A 'READ MOVIES' button is positioned above a search section that includes a text input field with 'Harry potter' and a 'Search' button.



When we insert a new movie with its information including: Name, Genre, Description, Language and Date, this data will be added into MongoDB database.

## Reading Information

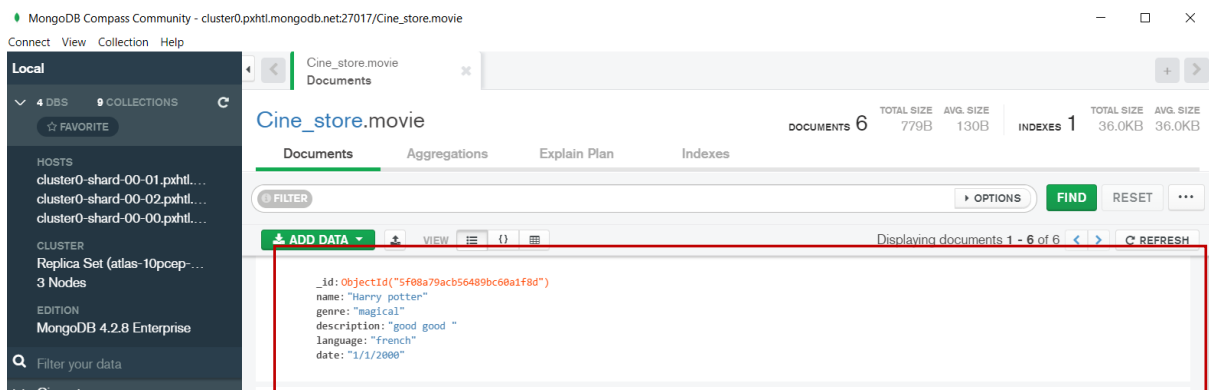




Once clicking on the Read Movies, the interface will show the list of all of film with related information. On the Reading session, we can use the search interface to find the information of a particular movie.

## Updating Information

This is the Database of “Harry Potter” before updating.



When we update the movie with the description: “Harry potter” with the Description: “ A movie from WB”.

The screenshot shows the Cine Store application interface. It has three main panels: **ADD MOVIE**, **DELETE MOVIE**, and **LIST ALL MOVIE**. The **ADD MOVIE** panel has input fields for Name, Genre, Description, Language, and Release Date, with an **ADD MOVIE** button at the bottom. The **DELETE MOVIE** panel has input fields for Name and Language, with a **DELETE MOVIE** button. The **LIST ALL MOVIE** panel has a large empty box for the movie list, a **READ MOVIES** button, and a search bar with a **Search** button. A fourth panel, **UPDATE MOVIE**, is highlighted with a red box. It contains input fields for Name (filled with 'Harry potter'), Description (filled with 'A movie from WB'), and Release Date (filled with '12/01/2000'), along with an **UPDATE MOVIE** button.

Then, the film “Harry potter” has been updated with the details “A movie from WB”

The screenshot shows the MongoDB Compass interface. The left sidebar shows the database structure with 4 DBS and 9 COLLECTIONS. The main panel displays the **Cine\_store.movie** collection. The **Documents** tab is active, showing a list of documents. The first document is highlighted with a red box, showing the following details: `_id: ObjectId("5f08a79acb56489bc60a1f8d")`, `name: "Harry potter"`, `genre: "magical"`, `description: "A movie from WB"`, `language: "french"`, and `date: "12/01/2000"`. The interface also shows statistics for the collection: 6 DOCUMENTS, 779B TOTAL SIZE, 130B AVG. SIZE, and 1 INDEXES.

## Deleting Information

We tried to delete the movie “Harry potter” by using Delete. The date of “Harry potter” was removed according my from the mongoDB.



Cine Store

ADD MOVIE

Name

Genre

Description

Language

Release Date

ADD MOVIE

DELETE MOVIE

Name

Language

DELETE MOVIE

UPDATE MOVIE

Name

Description

Release Date

UPDATE MOVIE

LIST ALL MOVIE

READ MOVIES

Search

MongoDB Compass Community - cluster0.pxhtl.mongodb.net:27017/Cine\_store.movie

Connect View Collection Help

Local

4 DBS 9 COLLECTIONS

FAVORITES

HOSTS

- cluster0-shard-00-01.pxhtl...
- cluster0-shard-00-02.pxhtl...
- cluster0-shard-00-00.pxhtl...

CLUSTER

Replica Set (atlas-10pcep-...)

3 Nodes

EDITION

MongoDB 4.2.8 Enterprise

Filter your data

- Cine\_store
  - movie
  - test
- admin
- config
- local

Cine\_store.movie Documents

DOCUMENTS 6 TOTAL SIZE 779B AVG. SIZE 130B INDEXES 1 TOTAL SIZE 36.0KB AVG. SIZE 36.0KB

Documents Aggregations Explain Plan Indexes

FILTER

ADD DATA VIEW

Displaying documents 1 - 5 of 5

REFRESH

```
{ "_id": ObjectId("5f08ac7e6aebbd8236fa777a"),  
  "name": "avenger",  
  "genre": "action",  
  "description": "great",  
  "language": "english",  
  "date": "9/21/2000"  
}
```

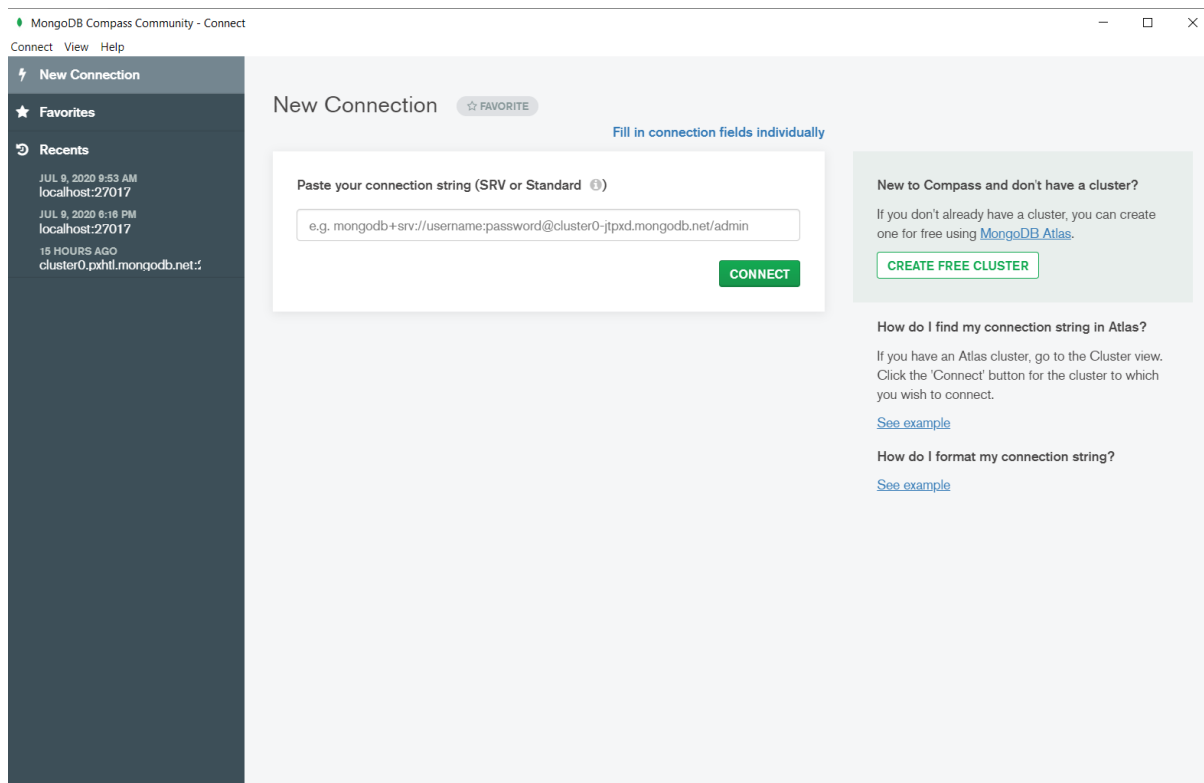
```
{ "_id": ObjectId("5f08ac9e6aebbd8236fa777b"),  
  "name": "tray vis",  
  "genre": "comic",  
  "description": "manga",  
  "language": "spanish",  
  "date": "9/21/2010"  
}
```

```
{ "_id": ObjectId("5f08b5c76a343b2755f26528"),  
  "name": "delicious"  
}
```

The database had removed the “Harry potter” movie.

# MongoDB

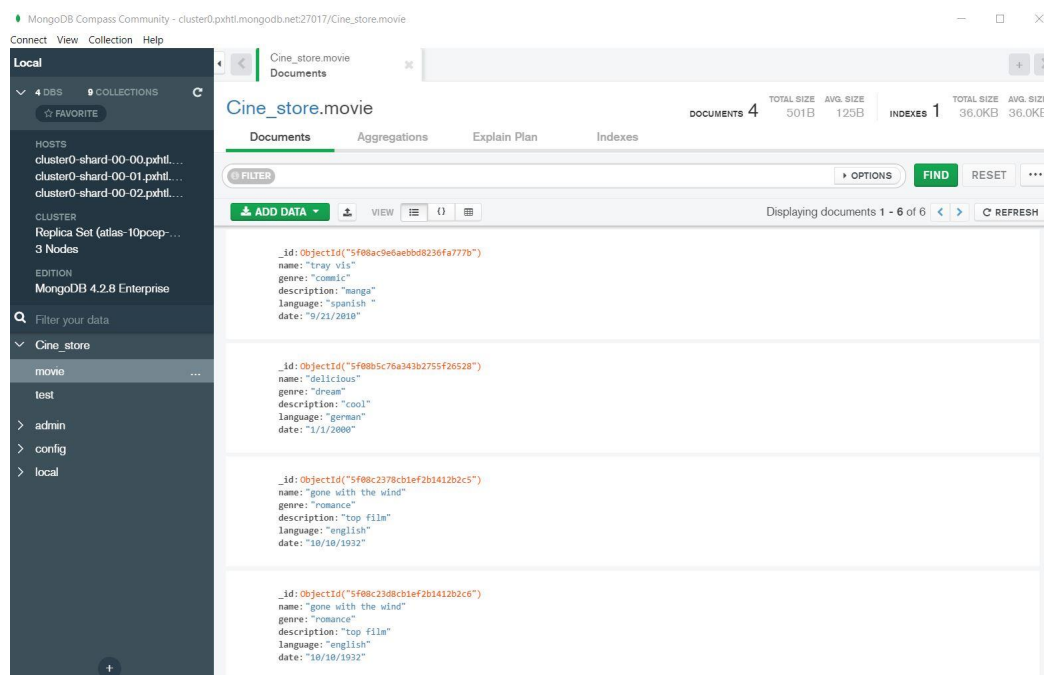
We used the newest version of MongoDB compass community which is 4.2



String connection that we have created:

mongodb+srv://junior Yao:epita123456789@cluster0.pxhtl.mongodb.net/test

Here is the Database of Cine\_store in MongoDB that we have used for the Movie application python based.



## The Python Coding

In order to connect the application interface with the MongoDB, we were requested to use PYMONGO, DNSPYTHON packages, below are the queries to the MongoDB server;

```
DatabaseManagement.py - C:\Users\junior\Desktop\MongoDB Project - Atom
File Edit View Selection Find Packages Help

DatabaseManagement..
import pymongo

connection = pymongo.MongoClient("mongodbsrv://juniorao:epita123456789@cluster0.mongodb.net/cdbname?retryWrites=true&majority")
db = connection['CineStore']
collection = db['movie']

def push_movie(name, genre, description, language, date):
    collection.insert_one({'name': name, 'genre': genre, 'description': description, 'language': language, 'date': date})

def delete_by_name(name, language):
    collection.remove({'name': name, 'language': language})

def display_all():
    assembler = []
    r = collection.find({})
    for i in r:
        assembler.append(i)
    return (assembler)

def search(name):
    assembler = []
    r = collection.find({'name': name})
    for i in r:
        assembler.append(i)
    return (assembler)

def update(name, description, date):
    collection.update_one({'name': name}, {'$set': {'description': description, 'date': date}})
```

```
final_app.py - C:\Users\junior\Desktop\MongoDB Project - Atom
File Edit View Selection Find Packages Help

DatabaseManagement.. final_app.py
self.pushButton_4.setText(_translate("CineStore", "READ MOVIES"))
self.pushButton.setText(_translate("CineStore", "Search"))

def addmovie(self):
    DatabaseManagement.push_movie(self.Movie_name_textedit.toPlainText(), self.Movie_genre_textedit.toPlainText(), self.Movie_description_textedit.toPlainText(), self.Movie_language_textedit.toPlainText(), self.dateedit.text())

def deletemovie(self):
    DatabaseManagement.delete_by_name(self.Delete_movie_name_textedit.toPlainText(), self.Delete_movie_language_textedit.toPlainText())

def updatemovie(self):
    DatabaseManagement.update(self.update_movie_name_textedit.toPlainText(), self.update_description_textedit.toPlainText(), self.update_dateedit.text())

def readallmovie(self):
    count = 1
    self.listallMovies.clear()
    assembler = DatabaseManagement.display_all()

    for i in assembler:
        for j, e in i.items():
            if j == "_id":
                self.listallMovies.insertItem(count, " ")
                count = count + 1
                continue

            self.listallMovies.insertItem(count, j + ": " + e)
            count = count + 1

def search(self):
    count = 1
    self.listView.clear()
    assembler = DatabaseManagement.search(self.textedit.toPlainText())

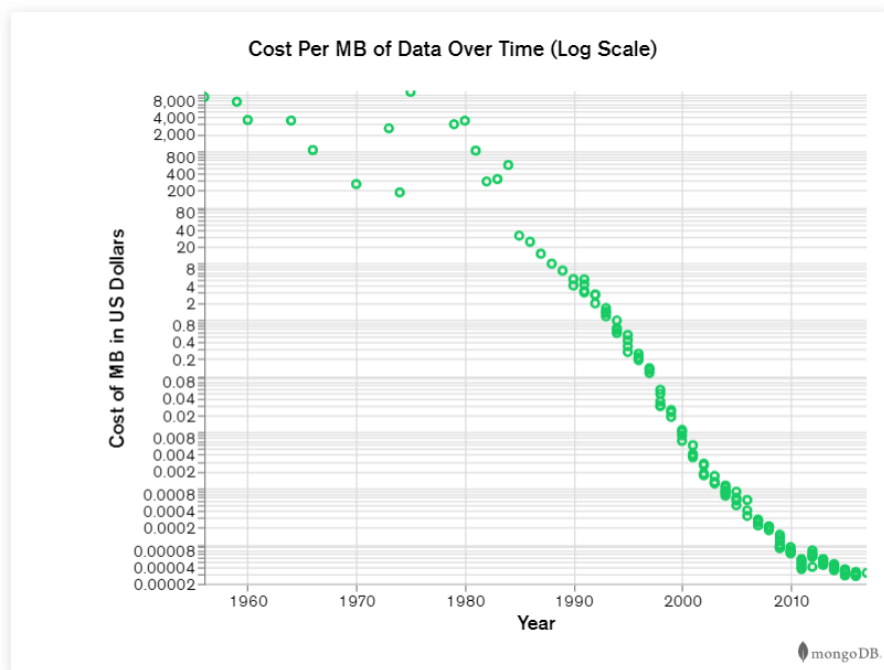
    for i in assembler:
        for j, e in i.items():
            if j == "_id":
                self.listView.insertItem(count, " ")
                count = count + 1
                continue

            self.listView.insertItem(count, j + ": " + e)
            count = count + 1
```

## Conclusion

NoSQL is a very powerful tool in database. Recently, the number of companies using NoSQL are increasing, from GAFAs to any start-up around the world. The project is just a humble demonstration to see the usefulness and effectiveness of MongoDB tool.

In fact, NoSQL databases emerged in the late 2000s as the cost of storage dramatically decreased. Gone were the days of needing to create a complex, difficult-to-manage data model simply for the purposes of reducing data duplication. Developers (rather than storage) were becoming the primary cost of software development, so NoSQL databases optimized for developer productivity.



## Reference

1. <https://en.wikipedia.org/wiki/NoSQL>
2. <https://www.mongodb.com/nosql-explained>