**Justin Branam**

**02-01-2021**

**CS 499**

**5-2 Milestone Four**

1. **Using the mongoimport tool, load the database “emails” with documents found in the “enron.json” file into the “enron” collection. Verify your load by issuing the following query:**

../startMongod.sh **#first step will always be to start mongo**

cd datasets/ **#change into the directory with the datasets**

mongoimport --db emails --collection enron ./enron.json **#mongo import utility**

mongo **# Type mongo to open the shell**

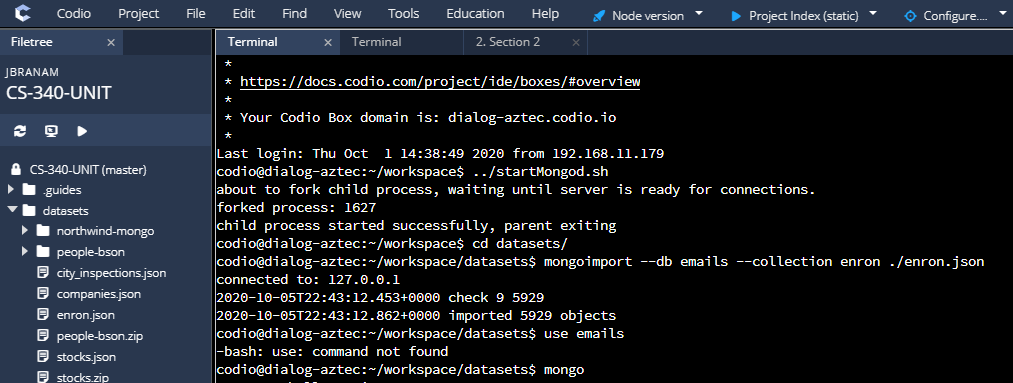
use emails **# Changes to the “emails” db**

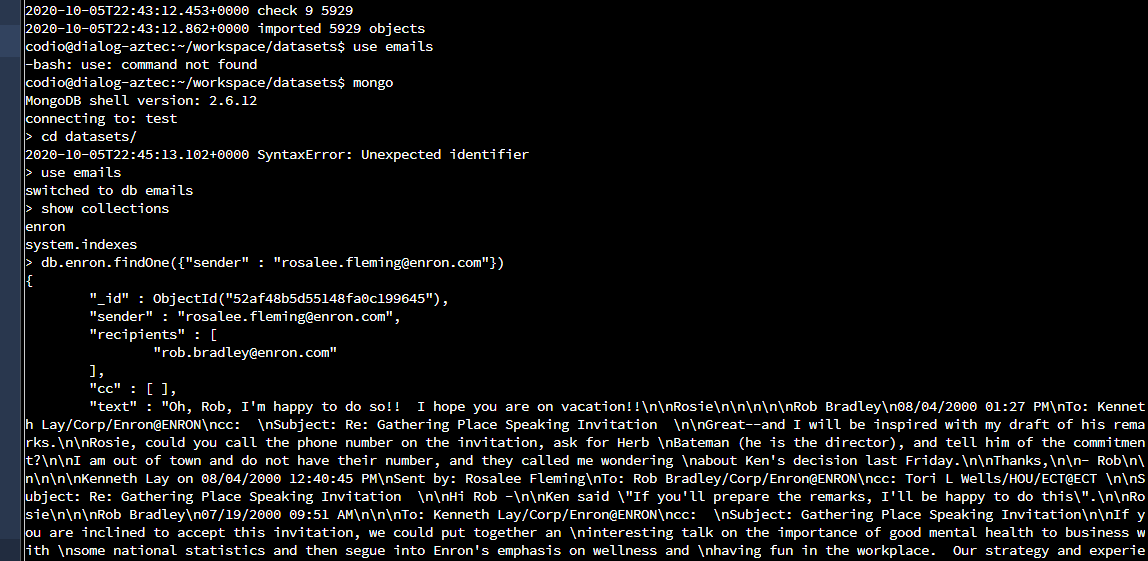
show collections **# Shows the enron collection**

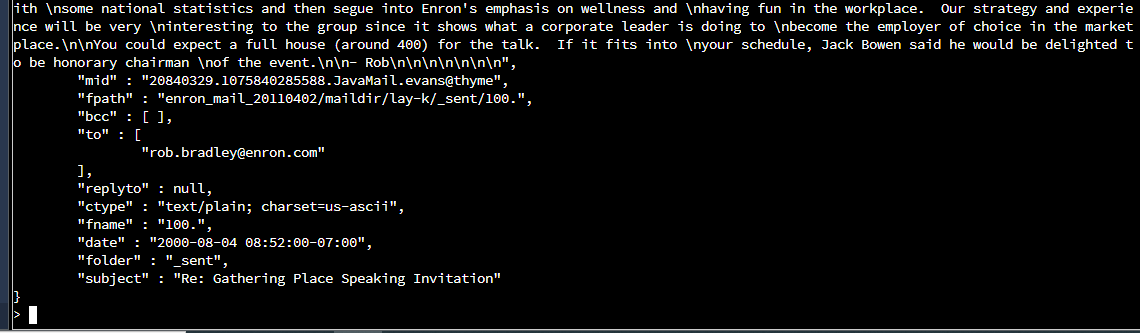
mongo **# Type mongo to open the shell**

db.enron.findOne({"sender" : "rosalee.fleming@enron.com"})

**Insert Screenshot:**

****

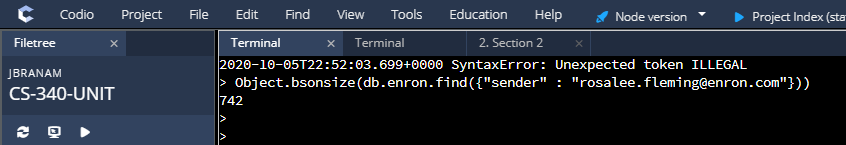
****

****

1. **Answer the following questions using MongoDB queries:**
2. What size is the document from Step 1?

Object.bsonsize(db.enron.find({“sender” : [rosalee.fleming@enron.com”})](mailto:rosalee.fleming@enron.com)) **# Will show size of the document**

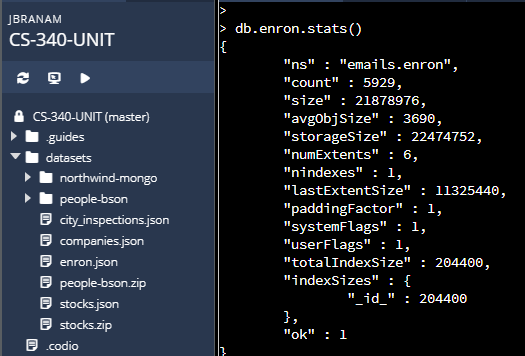
**Insert Screenshot:**

****

1. What size is the “enron” collection?

db.enron.stats() **# Shows size and details of the document**

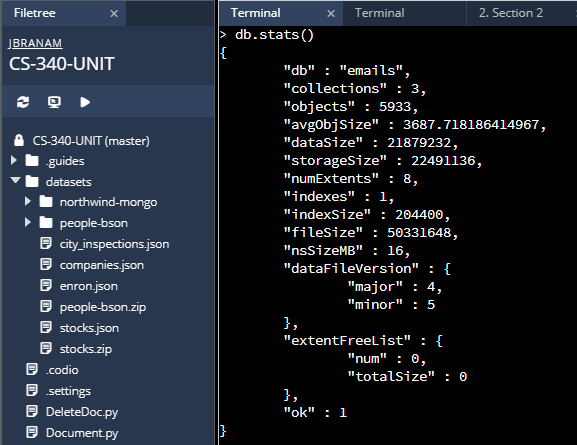
**Insert Screenshot:**

****

1. What size is the “emails” database?

db.stats() **# shows the size of the emails database and details**

**Insert Screenshot:**

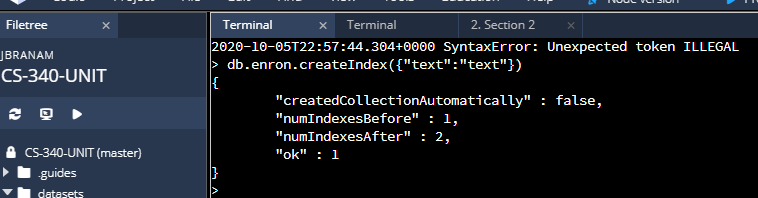
****

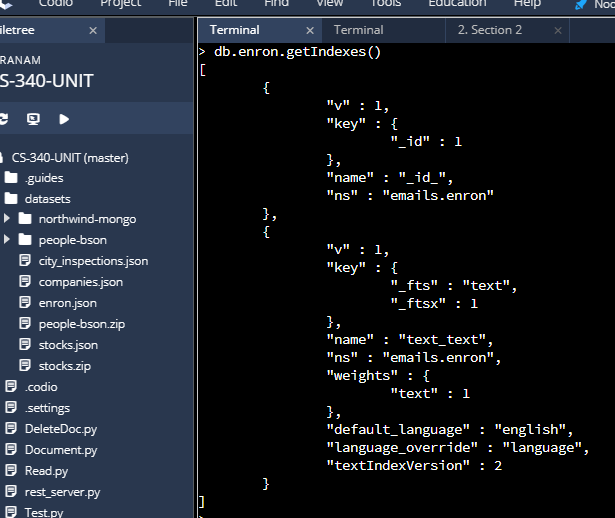
1. **Create a full-text index on the “text” key for the “enron” collection. Verify your index creation by providing screenshots of the list of indexes.**

db.enron.createIndex({“text”:”text”}) **# Creates the index “text” in the enron collection**

db.enron.getIndexes() **# Gets the text index in the enron collection**

**Insert Screenshot:**

****

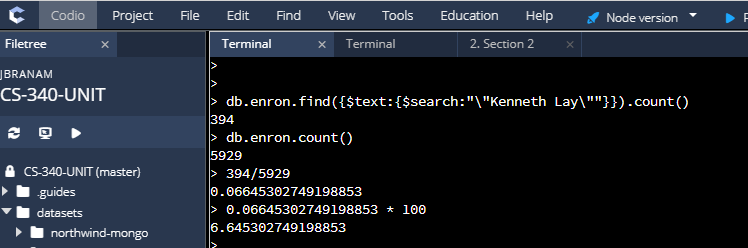
****

1. **What is the percentage of emails containing the string “Kenneth Lay” in the “enron” collection? Show your query as well as the answer.**

db.enron.find({$text:{$search:”\”Kenneth Lay\””}}).count()

db.enron.count() **# Not sure if this it right, show the total number of documents with Kenneth Lay and the total number of documents in the enron db. I then divide the number of Kenneth Lay documents by the total enron documents and multiply that by 100 to get the percentage.**

**Insert Screenshot:**

****

1. **Utilize the mongoimport tool to create a database named “market” and a collection named “stocks,” loaded with documents from the stocks.json file. Provide screenshots of the statements and the results of their execution.**

**../startMongod.sh # Starting MondoDB**

**cd datasets # changing to the datasets**

**mongoimport –db market –collection stocks ./stocks.json #Importing and creating a db called market and a collection called stocks**

**mongo # starting the mongo shell**

**use market # Switching to the market database**

**show collections # shows the collections in the market database**

**db.stocks.findone() # shows the results of the newly created socks collection**

**Insert Screenshot:**

Text

Description automatically generated

Text

Description automatically generated

1. **Assess the need for indexing as you formulate queries and, using the MongoDB shell, create any needed single or compound indexes. Provide screenshots of the statements and the results of their execution.**

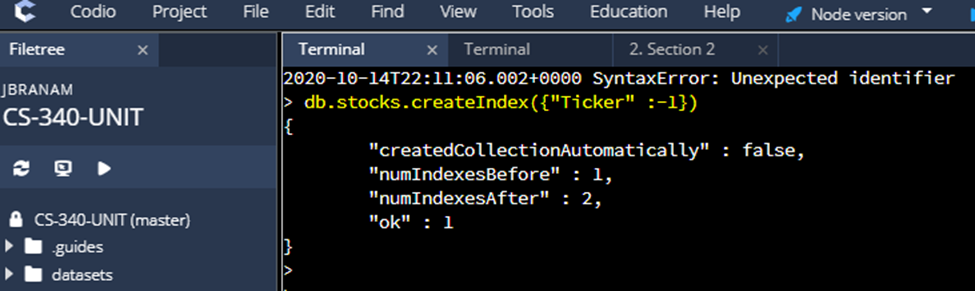
**Creating an Index:**

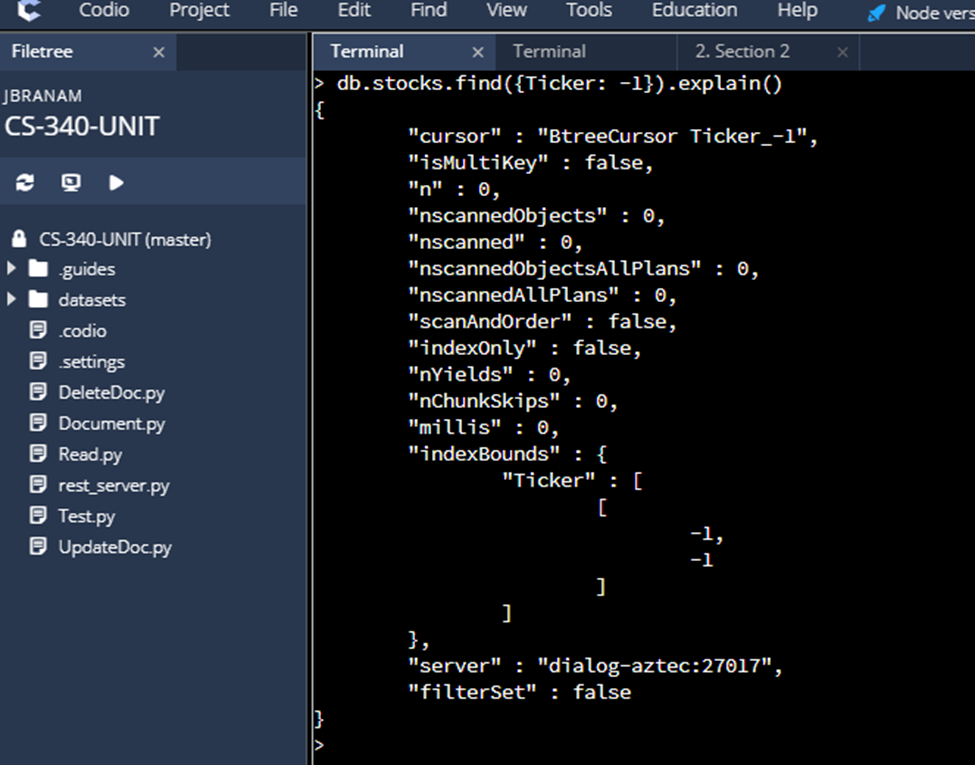
db.stocks.createIndex({Ticker:-1})

**Verifying newly created Index:**

db.stocks.find({Ticker: -1}).explain()

**insert Screenshot:**

****

****

1. **Develop a RESTful API using a Python or Java web services framework for a MongoDB collection of stock market summary data, ensuring your code is functional, reusable, concise, and commented.**

import json

from bson import json\_util

import bottle

from bottle import route, run, request, abort

import datetime

from pymongo import MongoClient

connection = MongoClient('localhost', 27017)

db = connection['market']

collection = db['stocks']

@route('/summaryinfo', method='GET')

def get\_summaryinfo():

tickers=request.query.tickers

print(tickers)

tickers=tickers.split(",")

findStr={"Ticker":{"$in":tickers}}

return read\_document(findStr)

def read\_document(document):

try:

result=collection.find(document)

return (json\_util.dumps(list(result)))

except Exception as ve:

abort(400, str(ve))

return False

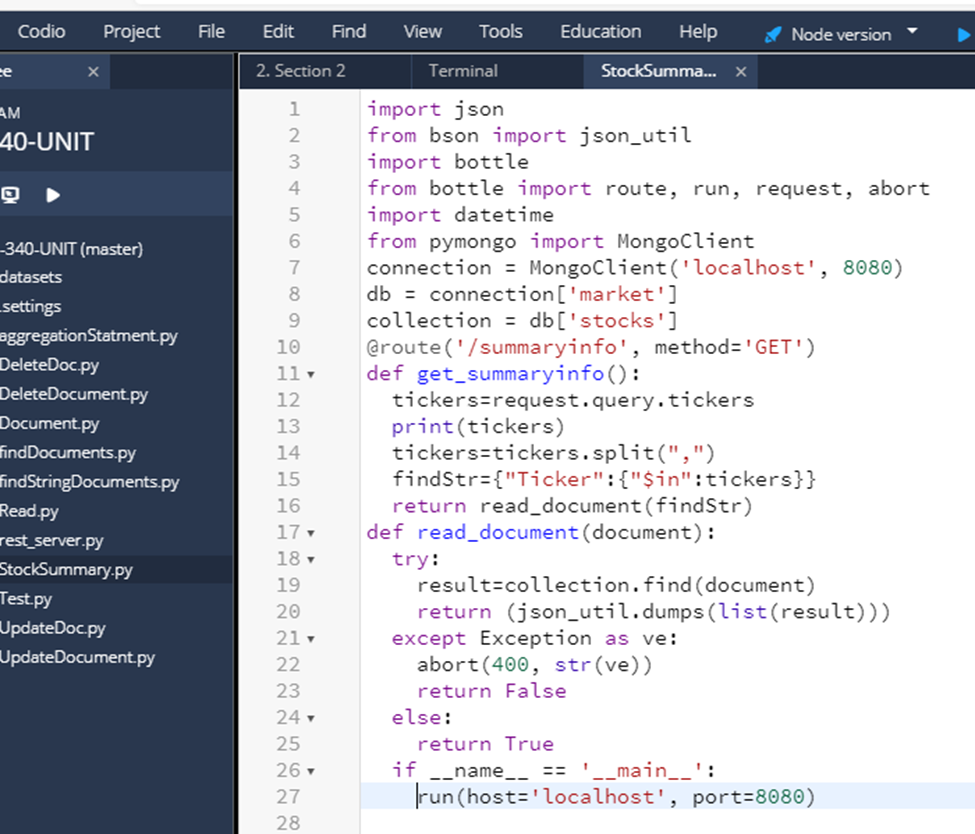
else:

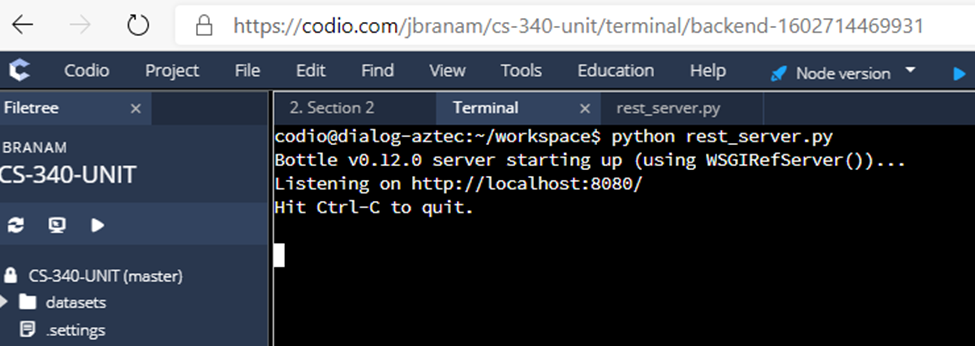
return True

if \_\_name\_\_ == '\_\_main\_\_':

run(host='localhost', port=8080)

**Insert Screenshot:**



****

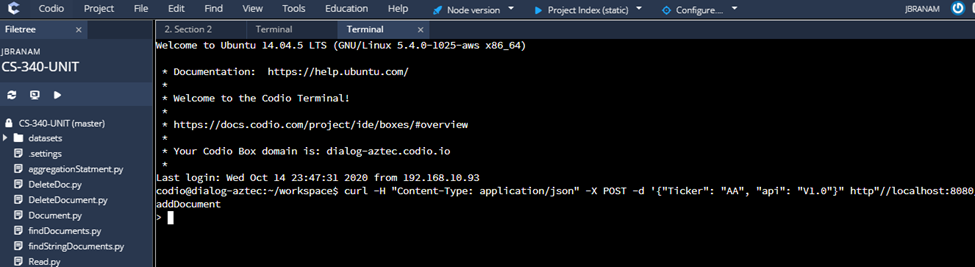
1. **Enable specific CRUD functionality in a developed RESTful API framework. Use the example URIs linked in the prompt to test and validate your framework. Provide screenshots of the code and its execution, ensuring your code is functional, reusable, concise, and commented.**

**In a new terminal window type:**

**For Create:**

curl -H “Content-Type: application/json” -X POST -d ‘{“Ticker”: “AA”, “api”: “V1.0”}’ http”//localhost:8080/addDocument

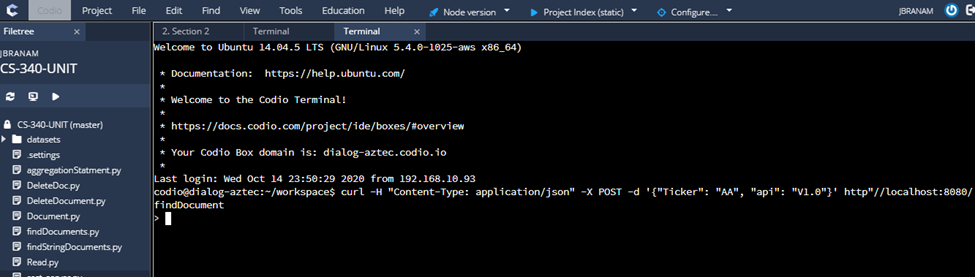
**Insert Screenshot:**

****

**For Read:**

curl -H “Content-Type: application/json” -X POST -d ‘{“Ticker”: “AA”, “api”: “V1.0”}’ http”//localhost:8080/findDocument

**Insert Screenshot:**

****