Introduction to MongoDB

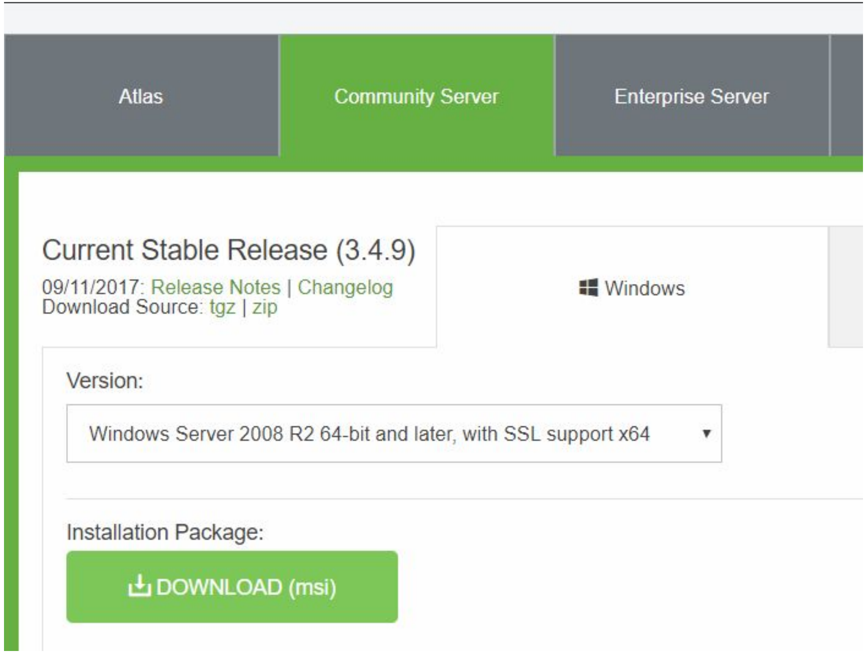
26/03/2018

What is MongoDB

- It’s a free NoSQL document oriented database.
  - Not only SQL, since SQL queries are supported.
- It uses JSON-like documents to save data.
  - Flexible schemas - Allows missing fields.
- High performance, high availability, and automatic scaling.
  - Used by large companies like Amazon.

Installation

MongoDB community server



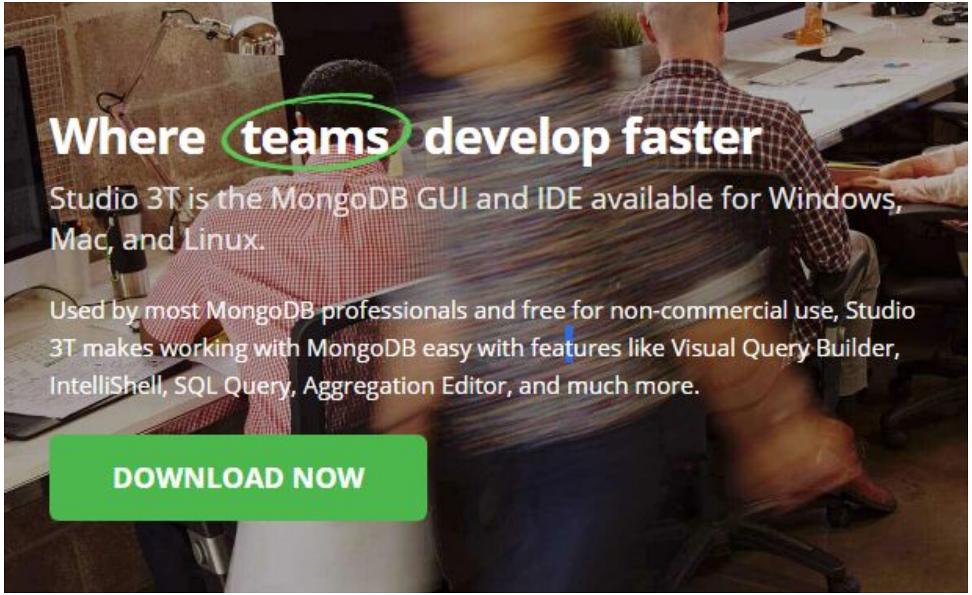
Current Stable Release (3.4.9)

09/11/2017: Release Notes | Changelog
Download Source: tgz | zip

Version: Windows Server 2008 R2 64-bit and later, with SSL support x64

Installation Package:

Studio 3T



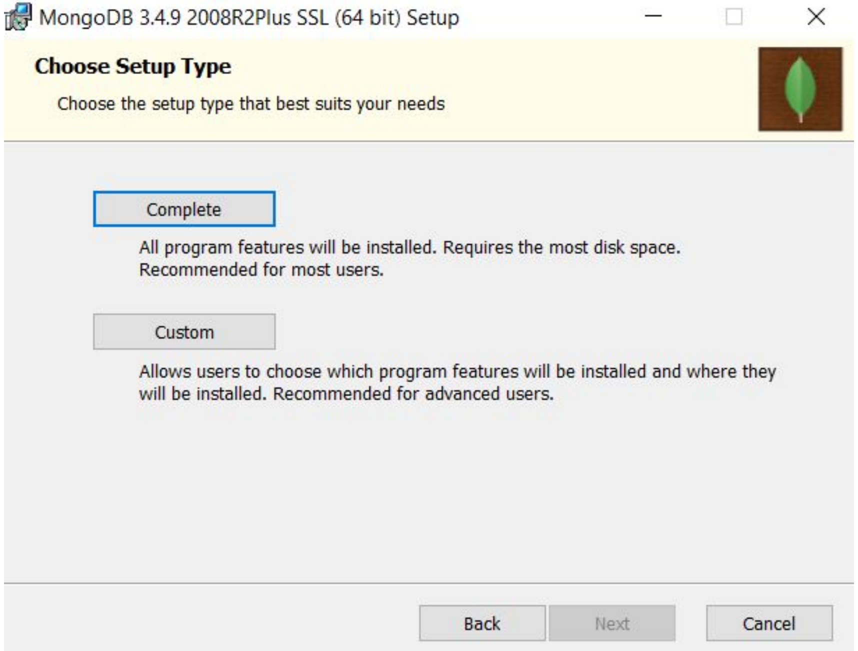
Where teams develop faster

Studio 3T is the MongoDB GUI and IDE available for Windows, Mac, and Linux.

Used by most MongoDB professionals and free for non-commercial use, Studio 3T makes working with MongoDB easy with features like Visual Query Builder, IntelliShell, SQL Query, Aggregation Editor, and much more.

Installation

Step 1



MongoDB 3.4.9 2008R2Plus SSL (64 bit) Setup

Choose Setup Type

Choose the setup type that best suits your needs

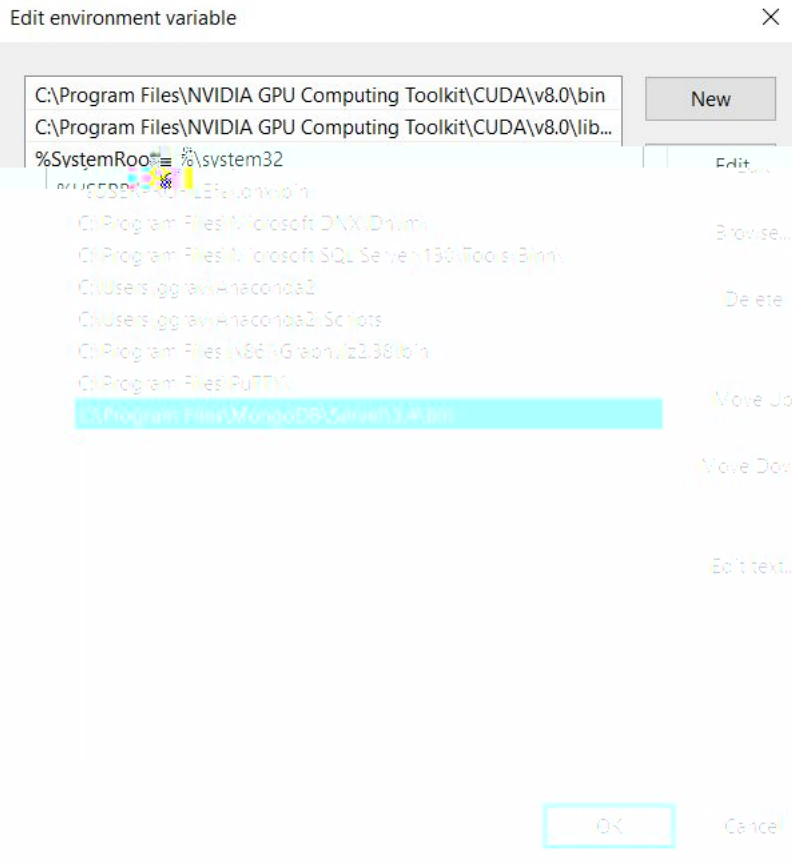
Complete

All program features will be installed. Requires the most disk space. Recommended for most users.

Custom

Allows users to choose which program features will be installed and where they will be installed. Recommended for advanced users.

Step 2

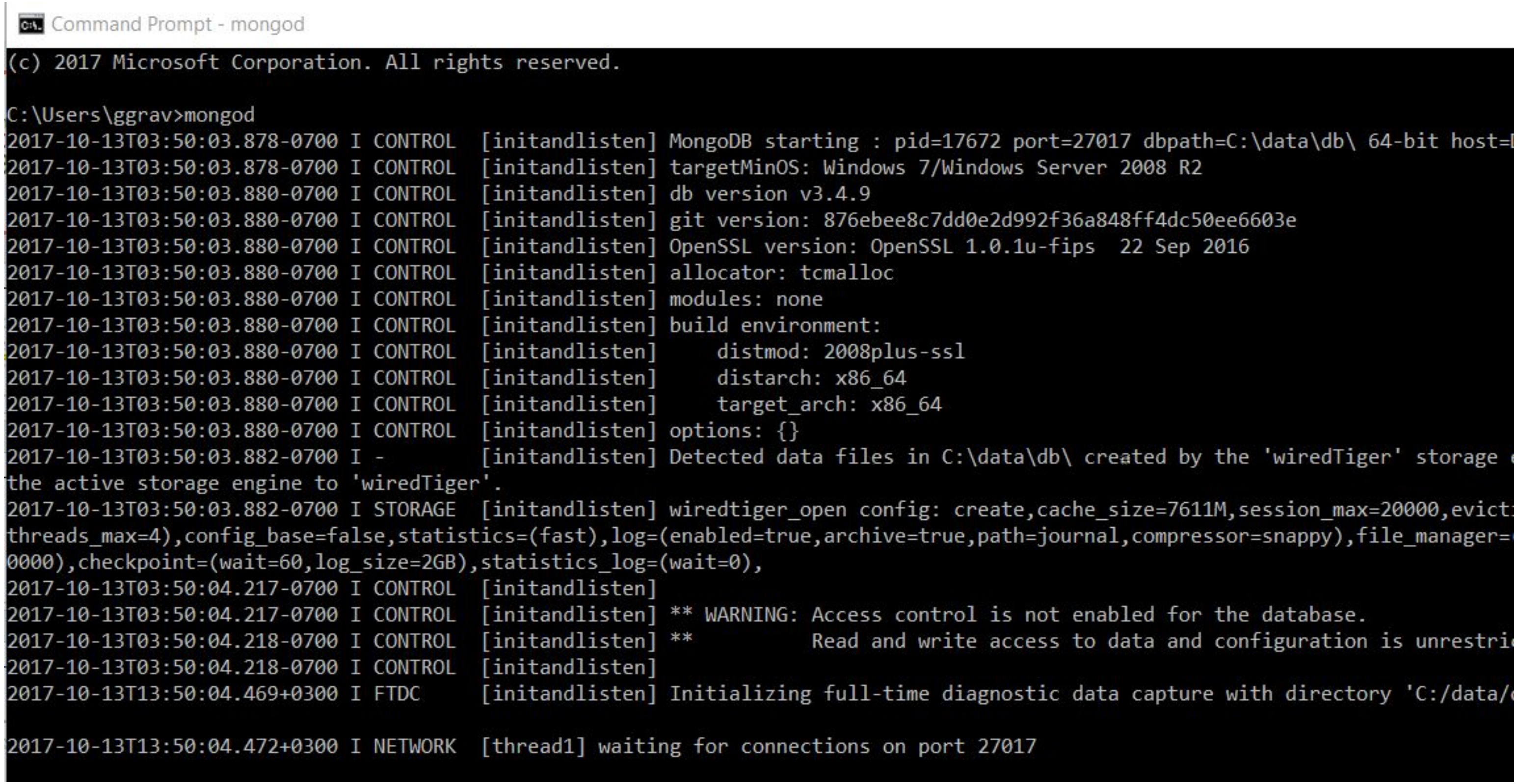


Edit environment variable

```
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v8.0\bin
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v8.0\lib...
%SystemRoot%\system32
%USERPROFILE%\.dnx\bin
C:\Program Files\Microsoft DNX\Dnvm\
C:\Program Files\Microsoft SQL Server\130\Tools\Binn\
C:\Users\ggrav\Anaconda2
C:\Users\ggrav\Anaconda2\Scripts
C:\Program Files (x86)\Graphviz2.38\bin
C:\Program Files\PuTTY\
C:\Program Files\MongoDB\Server\3.4\bin
```

Execution

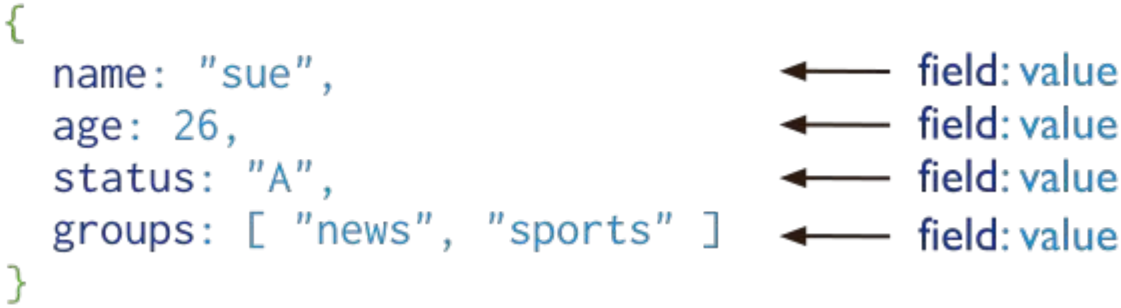
Step 3 -> Create folder “C:\data\db\” -> Open cmd -> run “mongod”



```
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\ggrav>mongod
2017-10-13T03:50:03.878-0700 I CONTROL  [initandlisten] MongoDB starting : pid=17672 port=27017 dbpath=C:\data\db\ 64-bit host=
2017-10-13T03:50:03.878-0700 I CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] db version v3.4.9
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] git version: 876ebee8c7dd0e2d992f36a848ff4dc50ee6603e
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] OpenSSL version: OpenSSL 1.0.1u-fips  22 Sep 2016
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] allocator: tcmalloc
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] modules: none
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] build environment:
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten]     distmod: 2008plus-ssl
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten]     distarch: x86_64
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten]     target_arch: x86_64
2017-10-13T03:50:03.880-0700 I CONTROL  [initandlisten] options: {}
2017-10-13T03:50:03.882-0700 I -        [initandlisten] Detected data files in C:\data\db\ created by the 'wiredTiger' storage
the active storage engine to 'wiredTiger'.
2017-10-13T03:50:03.882-0700 I STORAGE  [initandlisten] wiredtiger_open config: create,cache_size=7611M,session_max=20000,evict
threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=
0000),checkpoint=(wait=60,log_size=2GB),statistics_log=(wait=0),
2017-10-13T03:50:04.217-0700 I CONTROL  [initandlisten]
2017-10-13T03:50:04.217-0700 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2017-10-13T03:50:04.218-0700 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestri
2017-10-13T03:50:04.218-0700 I CONTROL  [initandlisten]
2017-10-13T13:50:04.469+0300 I FTDC     [initandlisten] Initializing full-time diagnostic data capture with directory 'C:/data/

2017-10-13T13:50:04.472+0300 I NETWORK  [thread1] waiting for connections on port 27017
```

What is a document

- A sum of key / value pairs



```
{
   name: "sue",
   age: 26,
   status: "A",
   groups: [ "news", "sports" ]
}
```

What is a collection

- A grouping of MongoDB **documents**.
- A **collection** is the equivalent of an RDBMS table.
- A **collection** exists within a single database.
- **Collections** do not enforce a schema.
- **Documents** within a collection can have different fields.
- **Collections** are grouped into **databases**.

Example: **db** = tweets, **collections** = greek_tweets, english_tweets, etc.

Operations

- Run / Connect to mongo
- Import documents
- Insert documents
- Create queries
- Quick data lookups
- Indexing

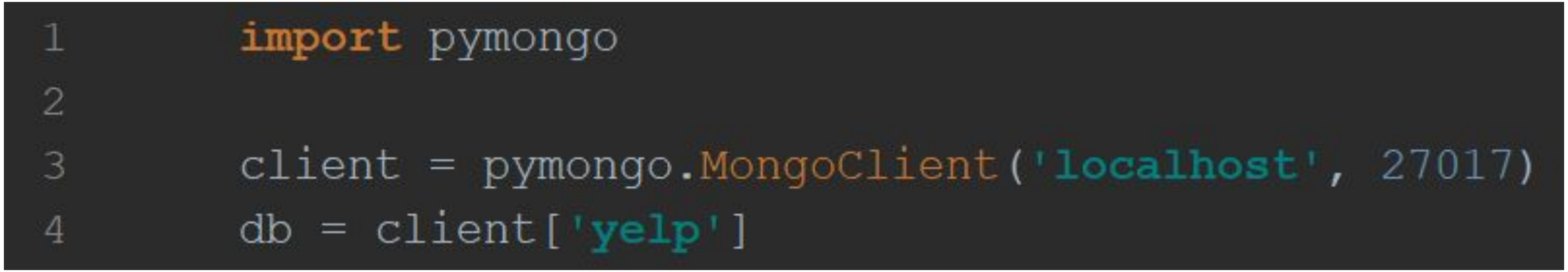
https://www.mongodb.com/what-is-mongodb

Connection

Studio 3T:

- Studio 3T -> Connect -> New connection -> (enter name) -> Save -> Connect
- Right click -> Add database -> (enter name) -> ok

Python:



```python
import pymongo

client = pymongo.MongoClient('localhost', 27017)
db = client['yelp']
```

Import dataset

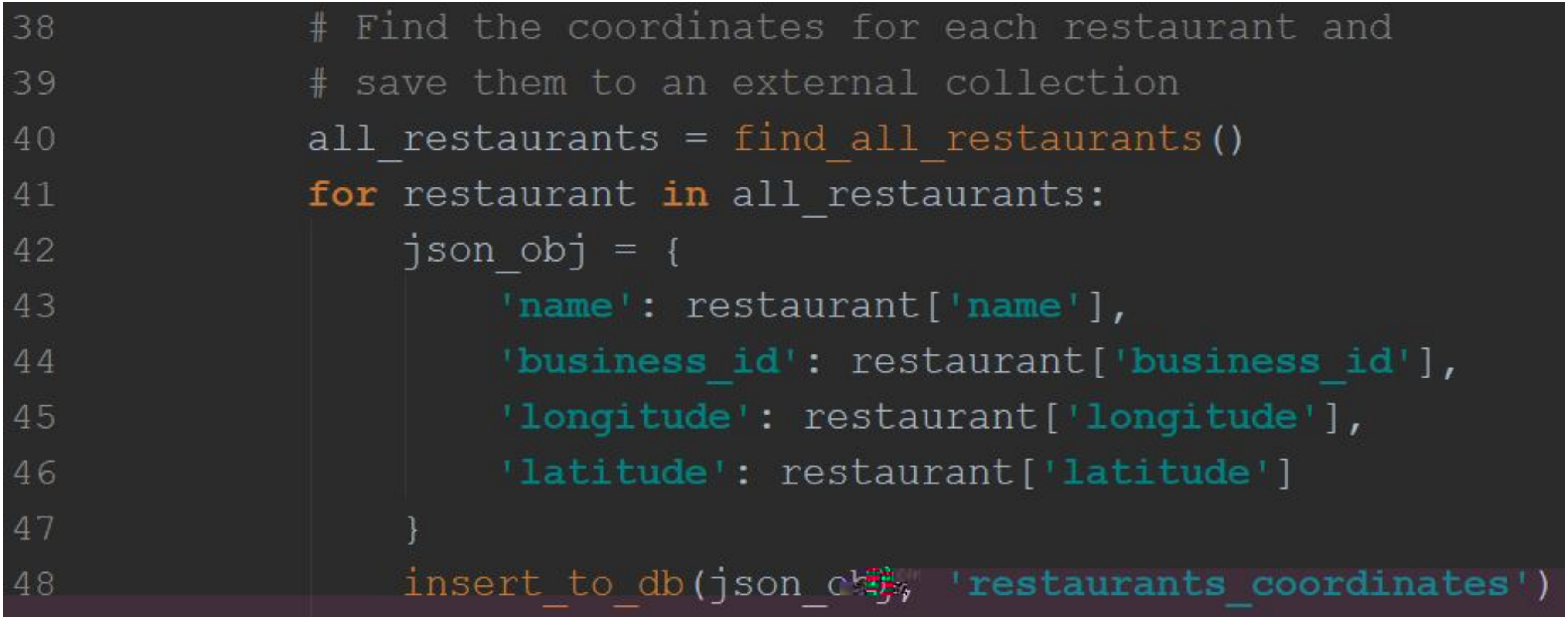
Studio 3T:

- Connect -> Select DB -> Import collections -> JSON -> Add sources -> Rename target collections -> Rename collections* -> Start import

* Rename collections to “restaurants”, “reviews” and “users”.

Insert documents

Python:



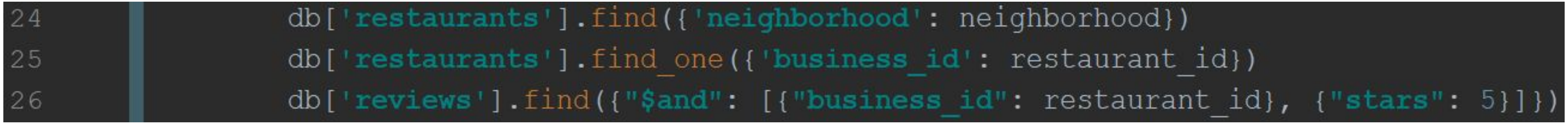
```python
# Find the coordinates for each restaurant and
# save them to an external collection
all_restaurants = find_all_restaurants()
for restaurant in all_restaurants:
    json_obj = {
        'name': restaurant['name'],
        'business_id': restaurant['business_id'],
        'longitude': restaurant['longitude'],
        'latitude': restaurant['latitude']
    }
    insert_to_db(json_obj, 'restaurants_coordinates')
```

Querying

Studio 3T:

- Right click collection -> Open Intellishell
- Examples:
  - db.restaurants.find({})
  - db.restaurants.find({"neighborhood": "Downtown"})
  - db.restaurants.find({ **$and**: [{"neighborhood": "Southeast"}, {"city": "Las Vegas"}]})

Python:



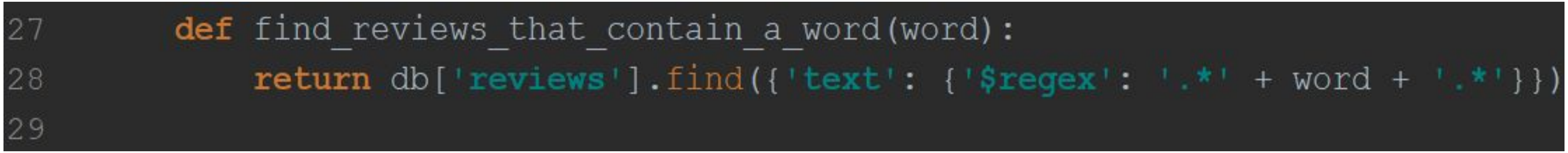
```
        db['restaurants'].find({'neighborhood': neighborhood})
        db['restaurants'].find_one({'business_id': restaurant_id})
        db['reviews'].find({"$and": [{"business_id": restaurant_id}, {"stars": 5}]})
```

Data lookups - For quick data checking

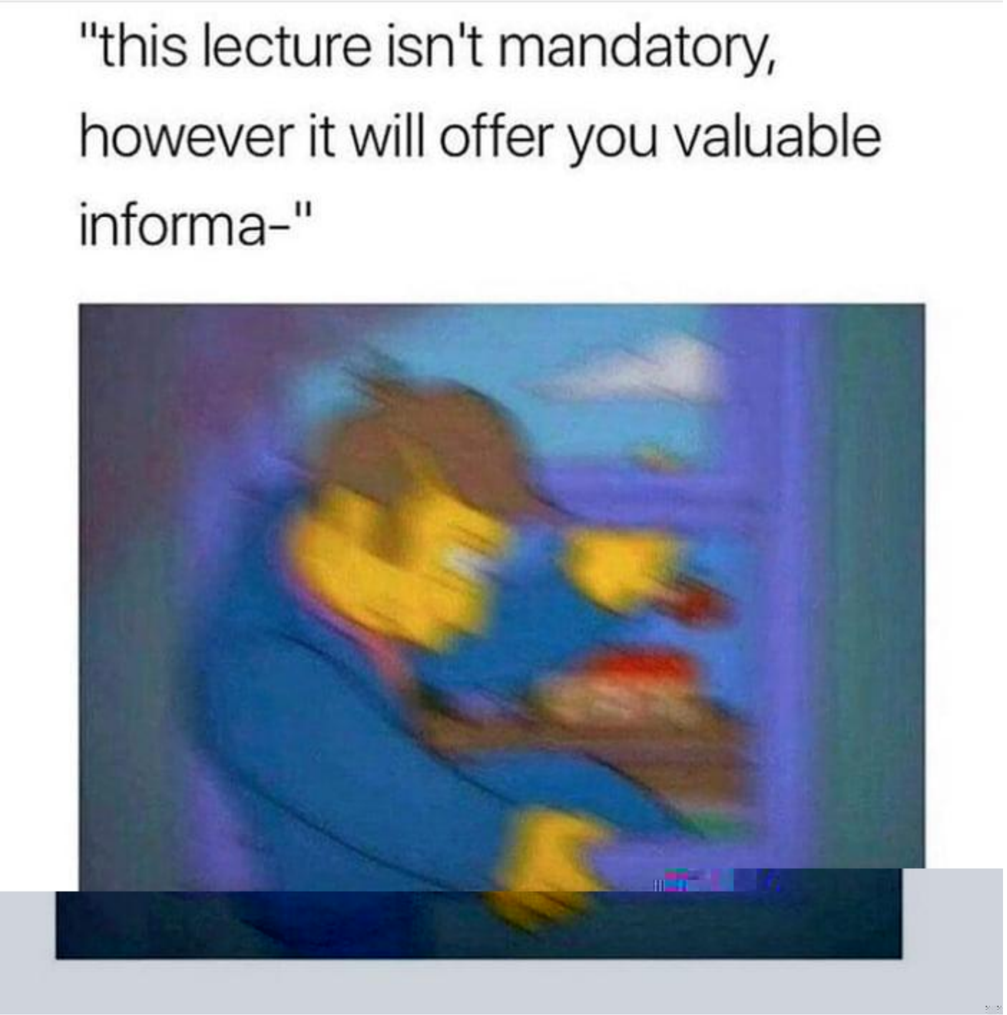
Studio 3T:

- Right click collection -> Open Intellishell
    - Ex: db.collection_name.find({ "Search_Field": "value" }, { "Field_to_display": 1 })
    - db.restaurants.find({ "neighborhood": "Downtown" }, { "name": 1 })
    - db.restaurants.find({ "name": /.*pollos.*/ }, { "text": 1 })
        - /.* is the regex equivalent for: “any single character, 0 or more times”

Python:



```
    def find_reviews_that_contain_a_word(word):
        return db['reviews'].find({'text': {'$regex': '.*' + word + '.*'}})

```



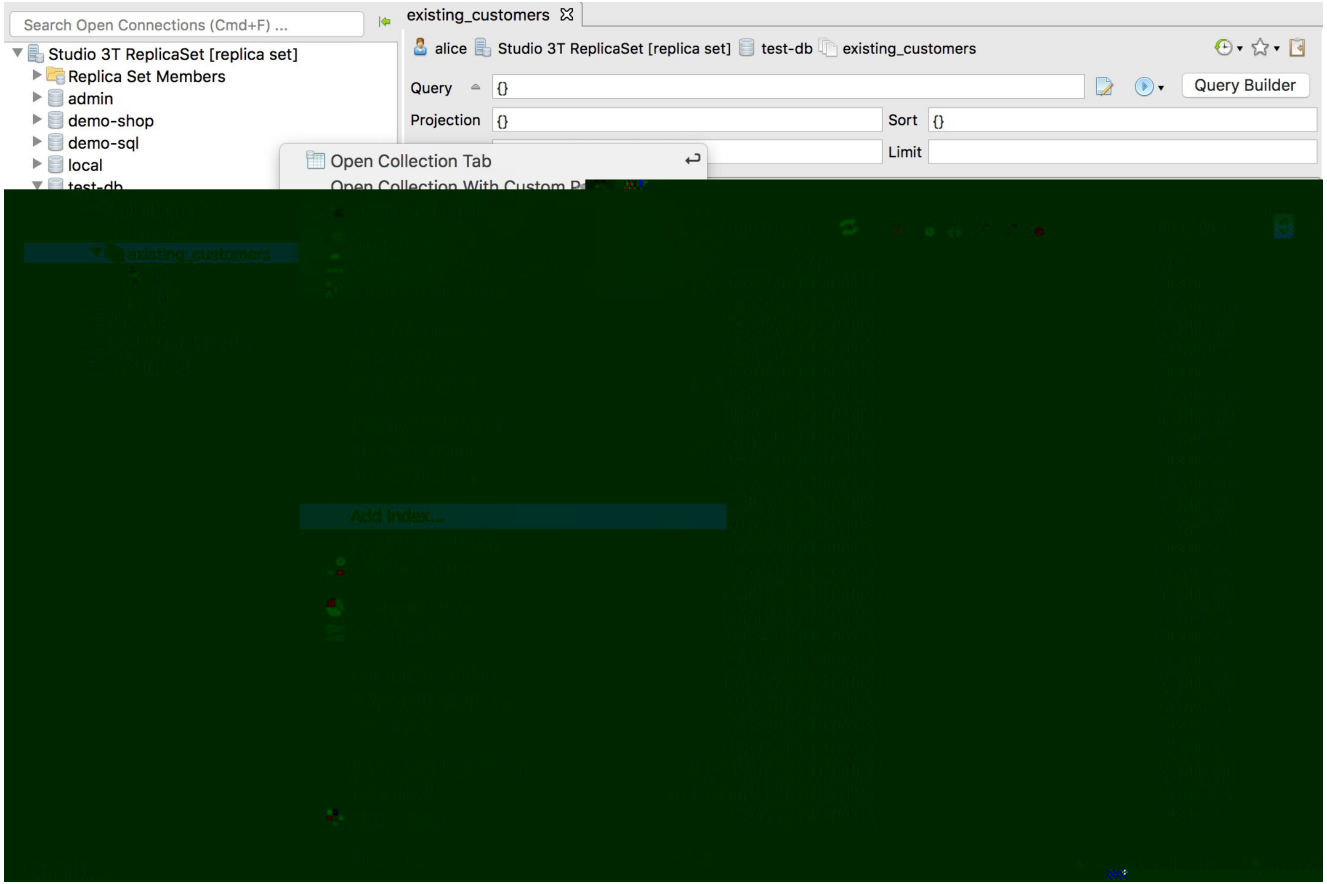
"this lecture isn't mandatory, however it will offer you valuable informa-"

Indexing

- Basic indexing
  - Speeds up queries on specific fields
  - Mostly used fields should be indexed
  - Index responsibly : Too many indices might slow down the database

- Text indexing
  - Faster text search queries on string content
  - Term lookup
  - Exact phrase lookup
  - Automatic relevance sorting

How to - Text indexing

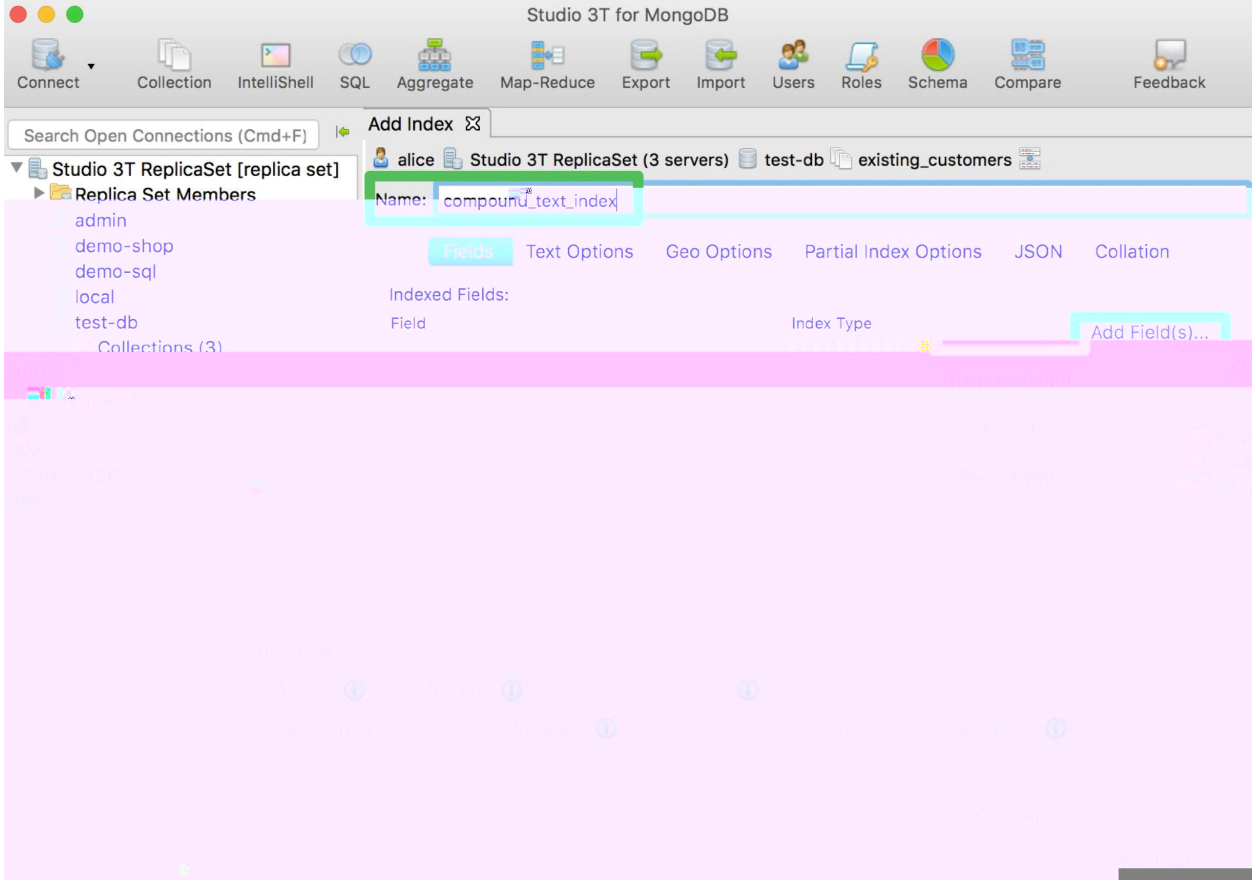


Query {}

Projection {}

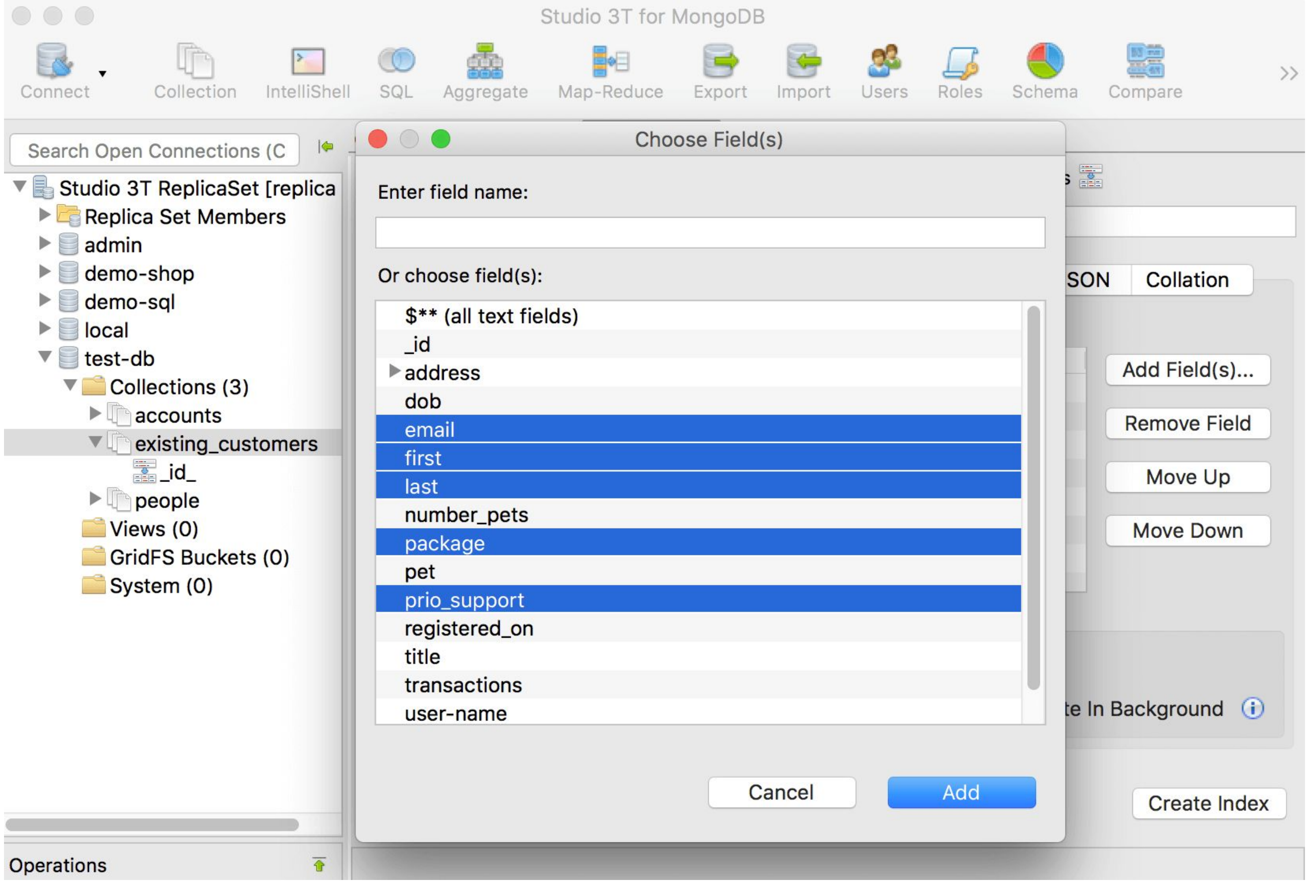
Sort {}

How to - Text indexing



Name: compound_text_index

How to - Text indexing



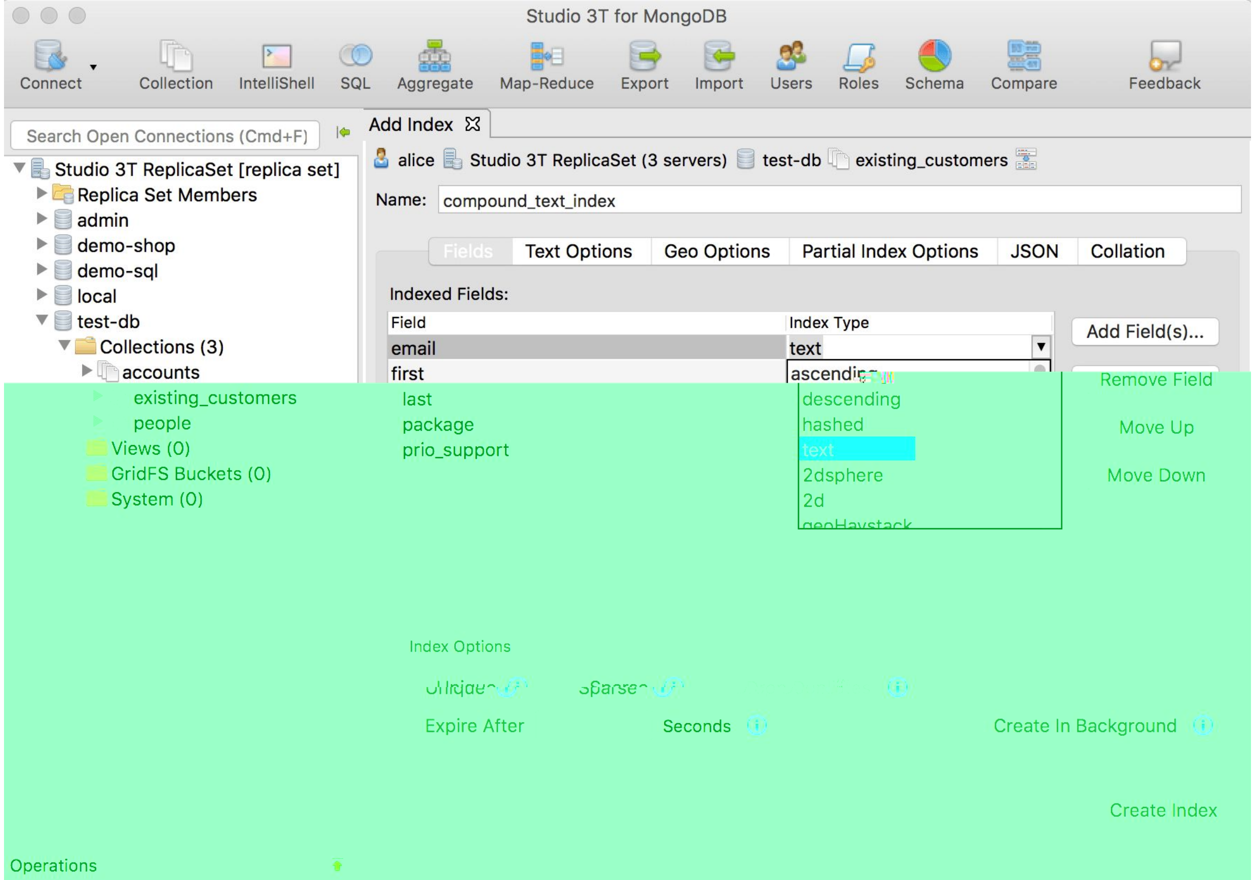
Choose Field(s)

Enter field name:

Or choose field(s):

- $** (all text fields)
- _id
- address
- dob
- email
- first
- last
- number_pets
- package
- pet
- prio_support
- registered_on
- title
- transactions
- user-name

How to - Text indexing

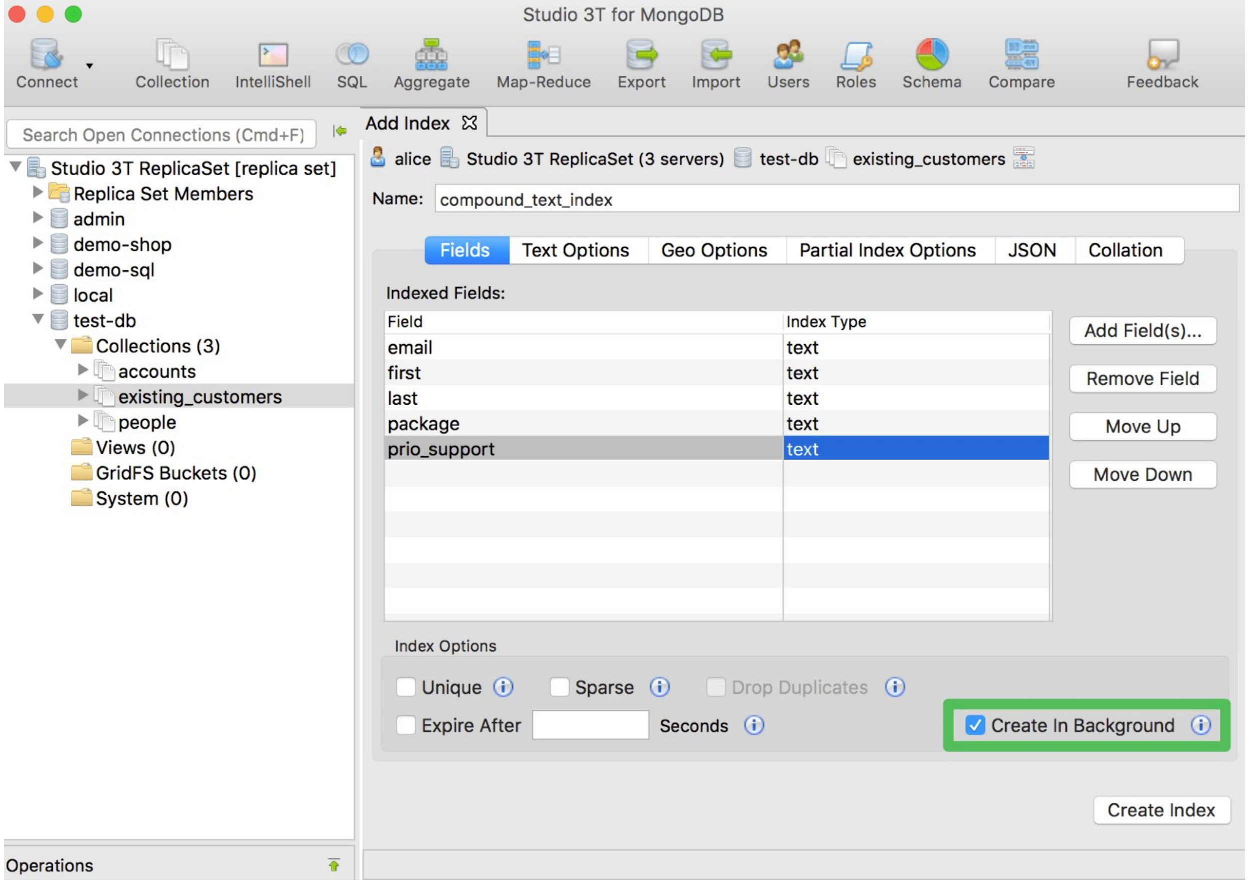


Name: compound_text_index

Indexed Fields:

Field	Index Type
email	text
first	ascending
last	
package	
prio_support	

How to - Text indexing



Name: compound_text_index

Indexed Fields:

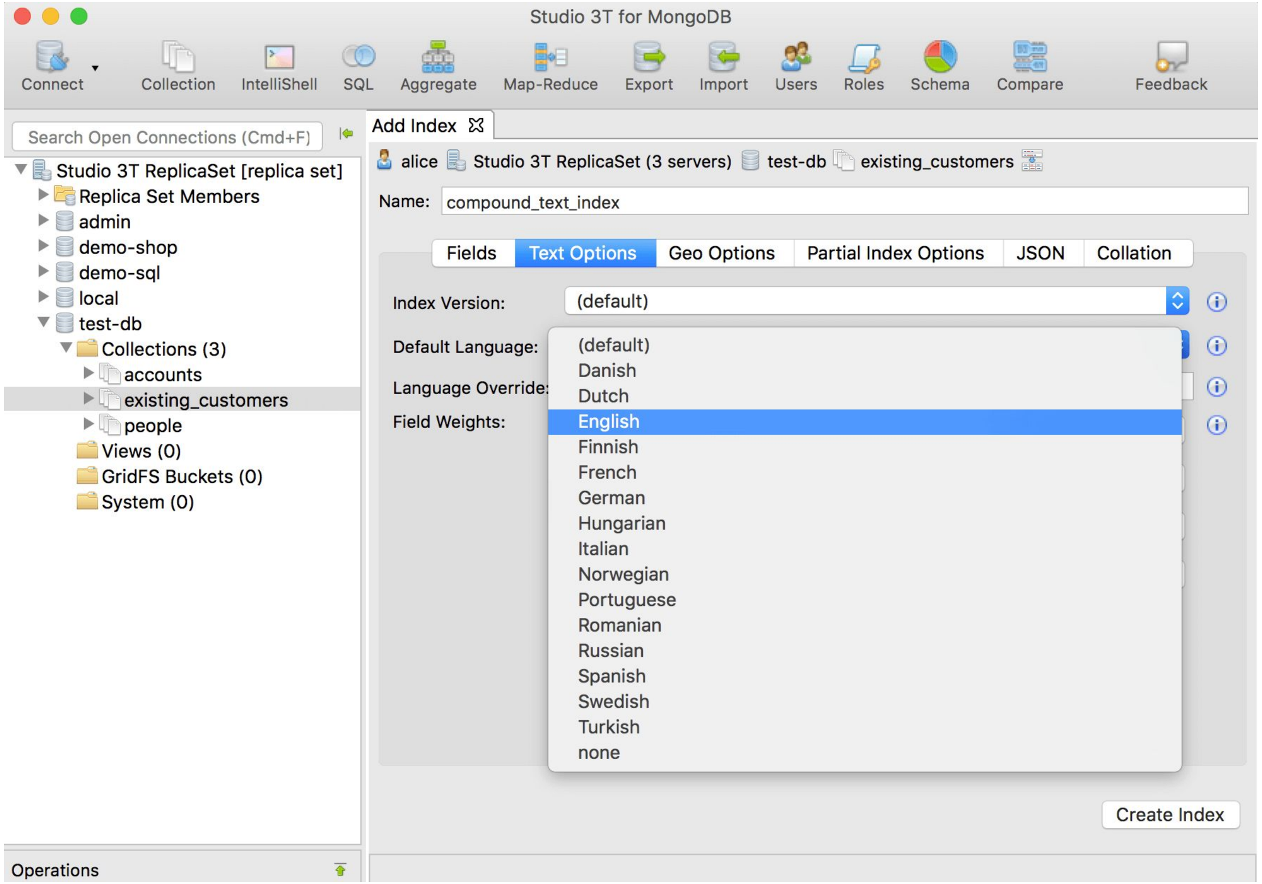
Field	Index Type
email	text
first	text
last	text
package	text
prio_support	text

Index Options

☐ Unique ☐ Sparse ☐ Drop Duplicates

☐ Expire After Seconds ☑ Create In Background

How to - Text indexing



Name: compound_text_index

Index Version: (default)

Default Language: English

Language Override:

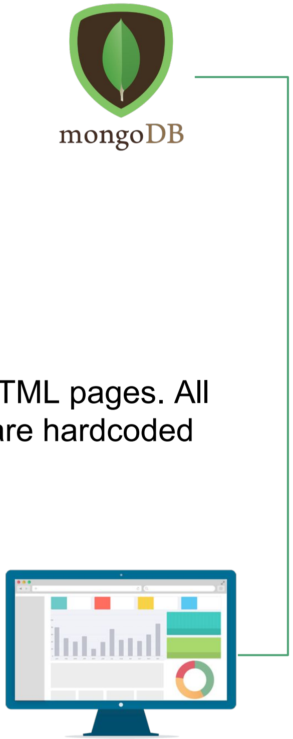
Field Weights:

Querying with text indices

- db.stores.find({ $text: { $search: "java coffee shop" } })
  - $text tokenizes the search string and performs a logical OR
  - Will search on all indexed fields
  - Results include a relevance score for each record
- db.stores.find({ $text: { $search: "java \"coffee shop\"" } })
  - Will match exact phrases “java” OR “coffee shop”
- db.stores.find({ $text: { $search: "java shop -coffee" } })
  - Will match (“java” OR “shop”) AND NOT(“coffee”)

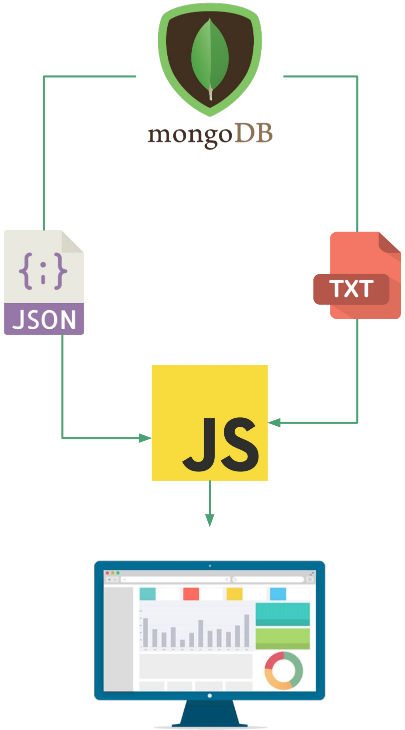
Web app development

Possibility #1

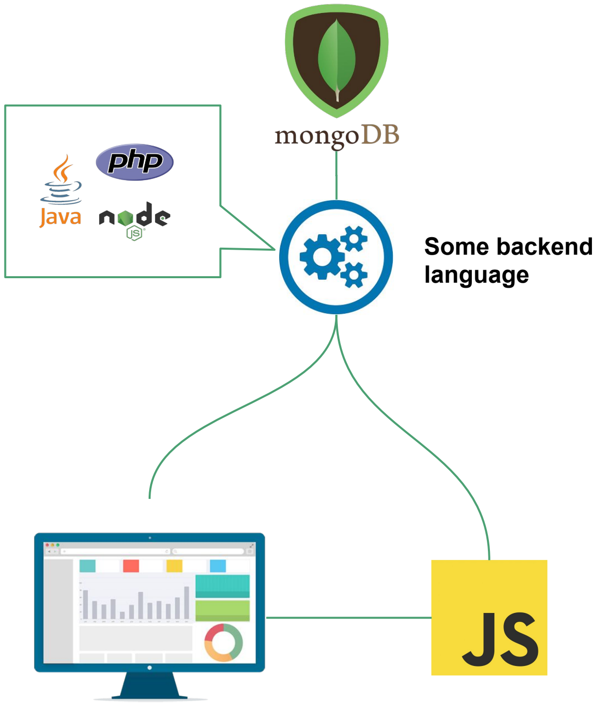


Static HTML pages. All results are hardcoded

Possibility #2



Possibility #3



Some backend language

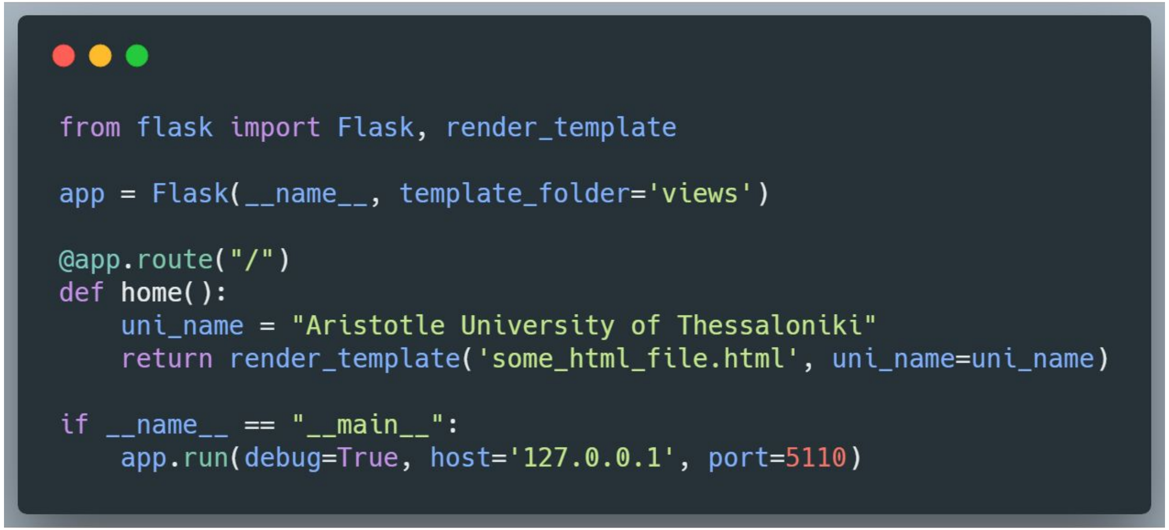
Flask

web development,
one drop at a time

A python-based microframework, suitable for small-scale applications.

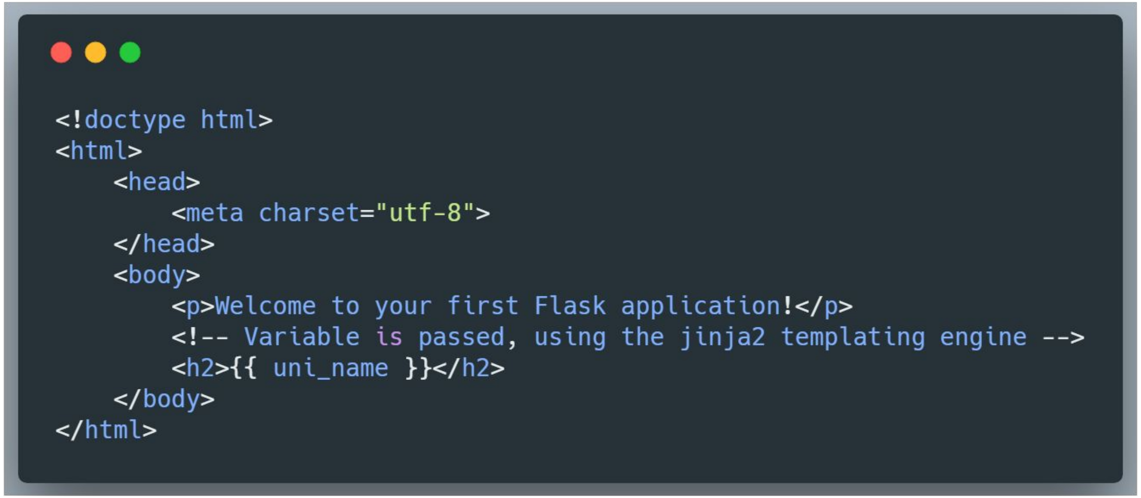
How to create a simple Flask App:

app.py

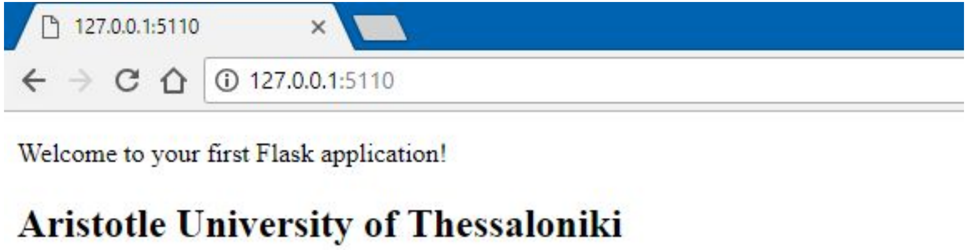


```python
from flask import Flask, render_template

app = Flask(__name__, template_folder='views')

@app.route("/")
def home():
    uni_name = "Aristotle University of Thessaloniki"
    return render_template('some_html_file.html', uni_name=uni_name)

if __name__ == "__main__":
    app.run(debug=True, host='127.0.0.1', port=5110)
```

some_html_file.html



```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
    </head>
    <body>
        <p>Welcome to your first Flask application!</p>
        <!-- Variable is passed, using the jinja2 templating engine -->
        <h2>{{ uni_name }}</h2>
    </body>
</html>
```



Welcome to your first Flask application!

Aristotle University of Thessaloniki

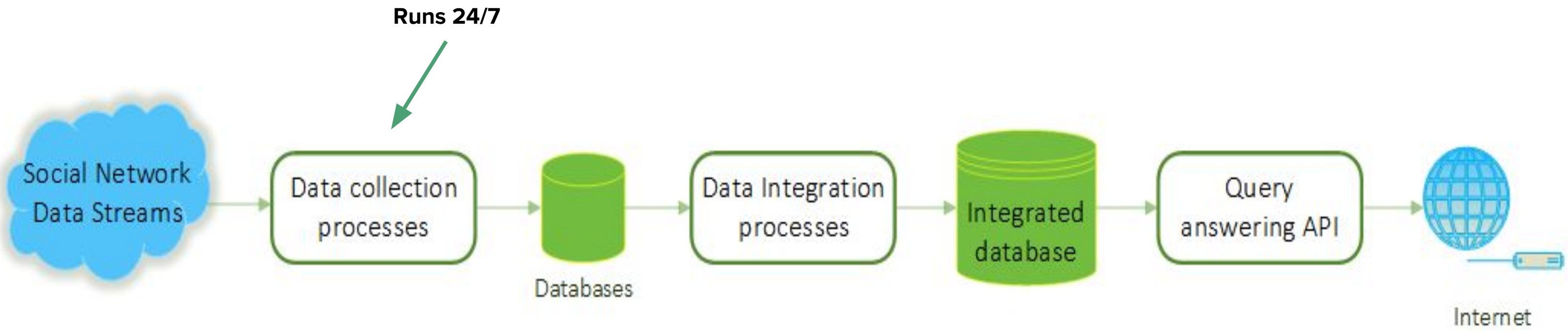
Diligent - A Social Media Data Integration platform

- Social media data explosion - Information Era
  - All data and metadata are saved into databases
  - Including private information (fingerprint, face, political views, etc.)

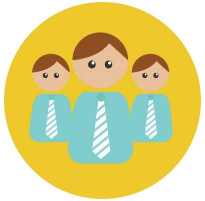
- Various data sources can be Integrated
  - Extract valuable insights
  - Personalized advertising
  - Sentiment analysis
  - -> Build a complete image - Increase data value $$

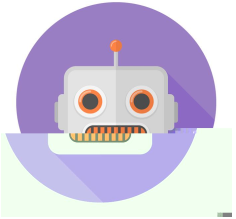
- Machine Learning opportunities
  - Integrated data as input

Diligent - System pipeline


Runs 24/7
Social Network Data Streams
Data collection processes
Databases
Data Integration processes
Integrated database
Query answering API
Internet

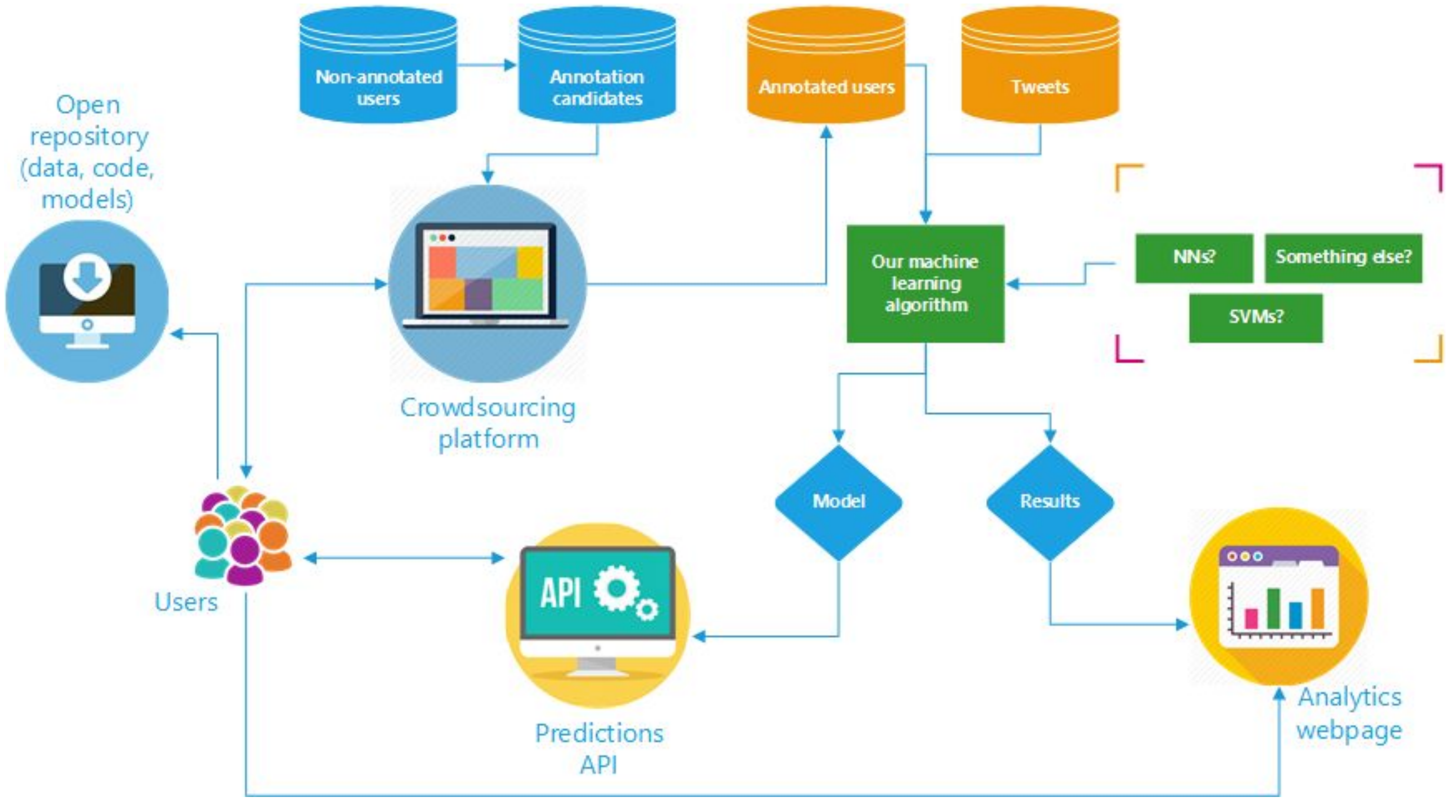
Twitter bots detection





- Own a set of pre-annotated users as bots or humans
- Extract features from each user (focus on sentiment)
  - E.g. “num of tweets”, “tweets entropy”, “average sentiment polarity”
- Feed a machine learning algorithm all the features vectors
- Evaluate and extract model

Full-scale architecture



Resources

- Dataset jsons
- Github link
- MongoDB
- Flask
- Flask-RESTful
- Python