# Setting up MongoDB

If you have a MongoDB server already running: jump to “Connecting to MongoDB”.

Else: do the following:

Install Mongodb

Download and execute the [mongodb-win32-x86\_64-2008plus-ssl-4.0.10-signed.msi](https://fastdl.mongodb.org/win32/mongodb-win32-x86_64-2008plus-ssl-4.0.10-signed.msi) file to download MongoDB. This download link can be found at [https://www.mongodb.com](https://www.mongodb.com/)

Add mongodb to Path variable

Search cortana for “edit the system environment variables”.

Click “Environment Variables”

Under the System variables, edit the “Path” variable and add the following directory:

C:\Program Files\MongoDB\Server\4.0\bin

Insert a dataset into mongo

Copy the information found [here](https://raw.githubusercontent.com/qmmr/mongodb/master/zips.json) into a text file and save it as zips.json.

In the directory you saved the file in, open a command prompt window and type in:

mongoimport –db examples --collection zips --file zips.json

Where examples is the name of the database and zips is the name of the collection.

# Connecting to MongoDB

To make a connection to the mongo database a python package needs to be downloaded which will act as the driver.

Install pymongo package with pip

To download a package, a package manager tool such as “pip” can be used. Package manager tools allow for easy installations of software.

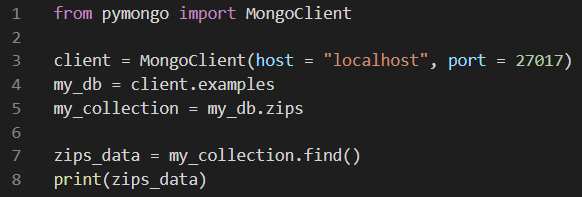
Open a command prompt and type in the command:

python -m pip install pymongo

The command will install the pymongo package to the computer and allow it to be used in all python files.

using pymongo

Create a new file and enter the following and save it as python\_mongo.py



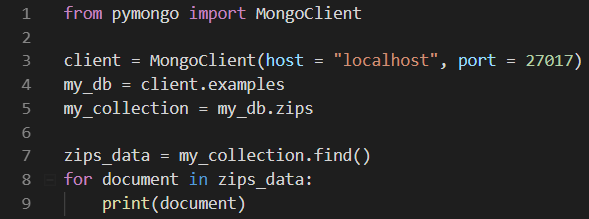
* Package: A collection of packages and modules
* Module: A python file. It will end in .py

MongoClient contains methods to interact with MongoDB. MongoClient is a class inside the mongo\_client module. The mongo\_client.py module, as well as many other modules are packaged into pymongo.

Therefore, from the pymongo package, MongoClient can be imported into our application.

We access the database and collection we want by using a dictionary-style access [as defined by the documentation](http://api.mongodb.com/python/current/api/pymongo/mongo_client.html). In this piece of code, there is a Mongo database called “examples” on the machine, and a collection of data inside that database called “zips”, [this was made during the setup](#_Setting_up_MongoDB).

Running the find() method retrieves all data within the collection. When zips\_data is printed the Class object name is shown at a memory location. To see the data inside this object, each document can be iterated over, and printed to the screen.



CRUD (CReate, read, update, delete) operations with pymongo

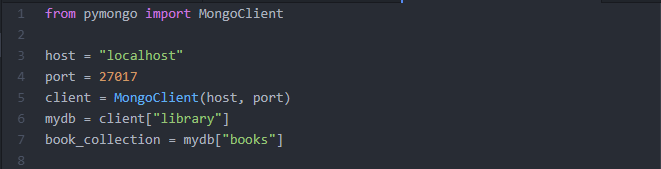
The pymongo commands are similar to the commands you used in the mongo shell, but snake case instead of camel case. So, findOne() becomes find\_one().

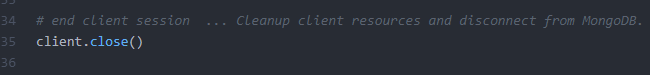
The conditions for finding a record are also similar to those you used in the mongo shell, for example find\_one({“title”: “Python Cookbook”}) will find a book with that title. For Read, Update and Delete, you have a filter that selects the documents you want to read, update or delete, and you get a return value that shows what happened.

For insert, you need to specify all the document data that you want to insert.

Each command is shown here in a simple case. There are many other options and return values that may be useful, check the pymongo documentation for details.

For all examples, you will make the connection to mongoDB and choose the library database and the books collection first, and close the client connection afterwards. (What happens if you don’t close the client?)

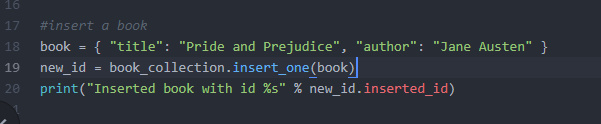




Insert a Record

Use insert\_one() or insert\_many()

Insert\_one()



Insert\_many()

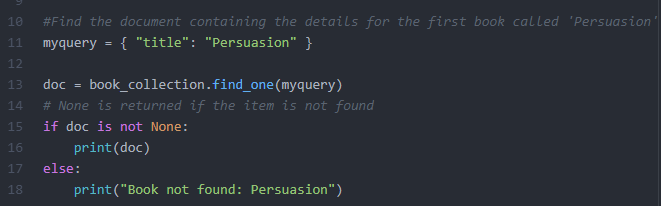


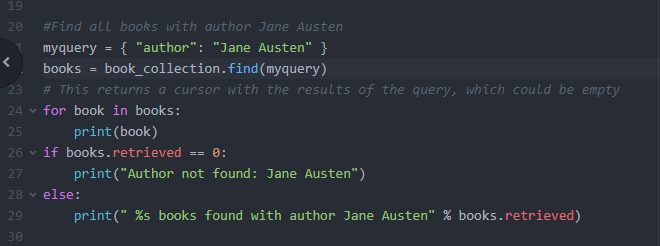
It would be sensible to check whether the book you are inserting already exists, you could do that using find\_one() or document\_count().

These examples set the data, but leave the document id to be set by mongoDB. You can set it yourself, but need to be sure it is unique, if you do.

REad a record

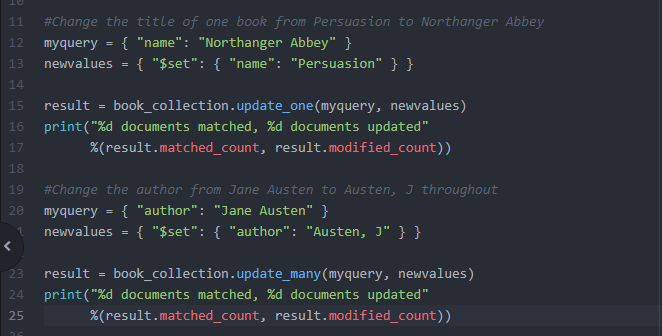
Use find\_one() or find() with a filter that shows what you want to find, as shown.





Update a record

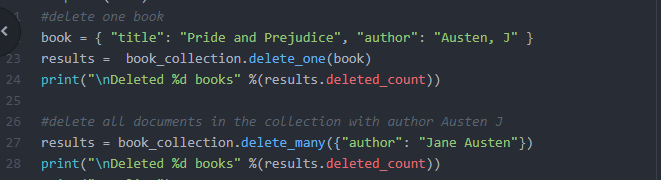
Use update\_one() or update\_many() with a filter that shows which documents you want to update, and a structure that shows what modifications you want to make.



The pymongo update commands return a results object with some useful fields, here matched\_count and modified\_count are used.

DELETE a record

Use delete\_one() or delete\_many() with a filter that shows which documents you want to delete. (It is easy to delete all documents in the collection with delete\_many!)



## Task 1 – Create as simple Store application CLI

A client TesGo has asked for a command line interface application to be used by the company to streamline the process of calculating funds, stock count etc.

User stories

As a Till Operator, I want to read all items and ids on the screen, so that I can see the id of each item in case the barcode doesn’t scan

As a Till Operator, I want to search for a specific item, so that I can quickly find the id of an item which doesn’t scan

As a Store manager, I want to add different items to the shop, so that a variety of products attracts customers

As a stock controller, I want to update the stock of any item so that items which have been broken can be discounted from the stock.

As a financial consultant, I want to update the cost of any item so that items can be on sale.

As a Store manager, I want to remove items from the shop, so that items which are dangerous will not be sold.

## Task 2 – Add Staff to the system

Be sure to fully finish Task 1 before starting task 2. The process of building a system, then adding functionality in quick iterations is the essence of Agile.

User stories

As a store manager, I want to add Staff to the system so that I can keep track of actions taken by which employee

As a store manager, I want to delete staff from the system so that fired employees can no longer have access to the records

As a customer assistant, I want to update my own staff records, so that I can change my surname in case I get married.

As a store manager, I want to see all Staff in my store so that I can remember their names.

As a store manager, I want to update staff roles so that I can give promotions

As a store manager, I want to update staff salary so that I can give financial incentives.

## Task 3 – Add purchases

**Create tests to make sure that all previous code is working**. For the next user stories, create the tests before you create the code (TDD).

User stories

As a store manager, I want to know which items are being purchased in a transaction, so that I can plan to get similar products.

As a customer, I want a receipt to show on screen after all my items have been scanned, so that I can identify if any items cost more than displayed on the shop floor.

As a customer, I want to be able to stop a transaction, so that if I forget my wallet at home I don’t have to pay.

As a store manager, I want to see which staff users have processed which purchases, so that I can see how fast each staff member is.

As a store manager, I want the “sold cost” to be separate from default cost, so that a history of sales will be accurate and won’t update when the default costs does.

As a store manager, I want to know the date and time of each purchase, so that purchases can be verified by the customer without having a paper receipt.

## Task 4 – Verification

For the following user stories, you will need to add verification. For instance, if a food item is added to the database without a sell-by date field, it will reject it being entered.

User stories

As a Health and Safety Officer, I want to make sure that stocked\_items which are in the food category must have a sell-by date so that food which cannot be sold can be searched for and removed

As a Stocker, I want all items to have a location which is the aisle it needs to go in, so that an item will always have a place to go.

As an Administrator, I want all added staff members to have a password of at least 6 characters, so that their accounts are secure

## Task 5 - Authentication

In this task you will need to authenticate yourself before you can start using the application.

Create tests before you code as this will make it quicker and easier to test what you are coding.

User stories

As a Staff member, I want to be able to login to the system, so that the work I do is attributed to me.

As a Staff member, I want to be able to logout of the system, so that when I am done no one uses my account

## Task 6 – Authorisation

User Stories

As a Store manager, I want each staff member to have roles so that I can allow only certain accesses to each user

As a Store manager, I want till operators to only be able to add purchases to the database

As a Store manager, I want store managers to only be able to add items to the database

As a store manager, I want Team leaders to only be able to update sales prices.

As a store manager, I want store managers to only be able to change roles of staff

As a store manager, I Want only team leaders and store managers to see and update the staff list.

As a store manager, I want only team leaders and stockers to be able to update stock of items.

## Task 7 – Automation

TesGo is open 24/7, and requires the same amount of people to work around the clock. The shifts are 9am-5pm, 5pm-1am and 1am-9am

User Stories

As a Team Leader, I want to be able to automatically generate the schedule for the next two weeks, so that I can let the employees know when they are in.

As a staff member, I want to be able to see when I’m scheduled to be in so that I don’t have to be constantly checking with my boss.

As a Team Leader, I want to set minimum contractual hours to staff members