# Self-studying project with MongoDB and PostgreSQL

This project aims to learning how to use Python work with MongoDB and PostgreSQL and use Flask to build a Web interface.

Technologies: Python packages: pymongo, sqlalchemy, flask, bokeh

Testing Data: the house prices estimation training data from Kaggle competition, [here](https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data).

Full training list:

* Insert data into MongoDB and PostgreSQL
* Query data from MongoDB and PostgreSQL
* Build Web interface

## How to set up server and database for web interface.

1. First, please download and install Mongodb, available [here](https://www.mongodb.com/download-center?jmp=nav#community) (under “Community server”) and Studio 3T, available [here](https://studio3t.com/download/?gclid=CjwKCAiA_c7UBRAjEiwApCZi8S22lBU81zoWG7zI8AAofJZpeBDKOUCDY-1J9EGkS-75mY6WHnFO3hoC4XUQAvD_BwE).
2. Set up database with Mangodump.

Run mango.exe in cmd.

Open Studio 3T and connect with DB server (localhost:27017)

In the Studio 3T, click the “Import” in the top graphical menu.

Choose BSON – mongodump folder -> select the mongodump folder -> make sure you check all collections in the test\_database -> --- -> Start Import

Now, you should have a database called test\_database under localhost:27017 DB server with 5 collections under Collections.

1. Run the test1.py in cmd.
2. Now that the server’s running, visit <http://localhost:8080/> with your Web browser. You’ll see the “House Price Estimation” web interface.

## Introduction of Web Interface

User with the web interface can do the following list of tasks:

1. Visualize example data table
2. Add new data into database
3. Searching data
4. Require data and download excel file
5. Require figures

*Structure of database:*

There are five collections in the database. They are “*test\_houseprices*”, the small amount of data for initial testing; “*currencyEuroBase*”, currency exchange rate information from [European Central Bank](http://www.ecb.europa.eu/stats/policy_and_exchange_rates/euro_reference_exchange_rates/html/index.en.html); “*data\_fields*”, data fields name and description; “*test*”, the full range of data; “*test\_quality*”, field name with missing data count and rate.

In the web interface example, there are five sections: “Example Data”, “Adding new price information”, “Searching for records”, “Exchange currency” and “Plot for Sale Price”, which are corresponding to the listed tasks, separately. The first four section only test with five fields from the original data as shown in the “Example Data”. There is a new column called “currency”, which is random generated from a currency list from “currencyEuroBase” collection.

1. “Example Data”, a static table shows the first 20 rows of the “test\_houseprices” collection. When the sale price is over 200000, the cell will be filled by red color.
2. “Adding new price information”, a html form can add new data into “test\_houseprices” collection. All four fields are required and house ‘Id’ will be automated generated.
3. “Searching for records”, a html form can request data from “test\_houseprices” collection with at least one limitation.
4. “Exchange currency”.
5. “Plot for Sale Price”.