To write using Raw HTML in a pug template, we can select the Pug option on the lower right corner of VSCode and switch it to HTML.

How to get data from pug template:

Add the following:

app.user(express.urlencoded())

app.post(‘/’,(req,res)=>{

console.log(req.body);

let name = req.body.name //name property in id name

})

‘/’ is the form action. Req.bosy has the data as json object

**Making a pug project from scratch:**

Create a folder and open VSCode in it

npm init -y

import the following:

const express = require(‘express’);

const path = require(‘path’);

const fs = require(‘fs’);//only if you want to use file for saving and reading data

const app = express();

const port = 80;//port can be changed

install express and pug

//EXPRESS SPECIFIC STUFF

app.use('/static',express.static('static'));//for serving static files

app.use(express.urlencoded());

//PUG SPECIFIC STUFF

app.set('view engine', 'pug');//set the template engine as pug

app.set('views', path.join(\_\_dirname, 'views'));//set the views directory

**Mongodb**

Install community server

To run Mongodb as a mac os service:

brew services start mongodb-community@7.0

To stop a [mongod](https://www.mongodb.com/docs/manual/reference/program/mongod/" \l "mongodb-binary-bin.mongod) running as a macOS service, use the following command as needed:

|  |
| --- |
| brew services stop mongodb-community@7.0 |

If you started MongoDB as a macOS service:

brew services list

Connect and Use MongoDB

To begin using MongoDB, connect mongosh to the running instance. From a new terminal, issue the following:

mongosh

To create a new database: use testDb (a new database will be created if it doesn’t already exists, with the name testDb).

//to insert a new document in items collection. db refers to the current database which is testDb, in this case. With this command a new collection ‘items’ will be created and a new document will be inserted into it.

db.items.insertOne({name:”item1”, price:22000, rating: 4.5, qty: 233, sold:98})

//to insert more than one

db.items.insertMany([{name:”item2”, price:22000, rating: 4.5, qty: 233, sold:98}, { name:”item3”, price:22000, rating: 4.5, qty: 233, sold:98}])

to see the documents in a collection

db.items.find()

mongo by default gives duplicate id to each document

mongodb doesn’t enforce a schema, so it is not necessary that all columns are the same while inserting data in a collection.

to view all collections in a db-> show collections

filtering data in mongodb

* db.items.find({price : 3.5}) //show all items with price=3.5, here query in curly brackets is called filter object
* db.items.find({rating : {$gte: 3.5}})//ratings greater than or equal to 3.5
* db.items.find({rating : {$gt: 3.5}})//ratings greater than 3.5
* db.items.find({rating : {$gt: 3.5}, {price:{$gt:4000}}})//ratings greater than 3.5 and price greater than 4000
* db.items.find({rating : {$lt: 3.5}, {price:{$gt:4000}}})//ratings less than 3.5 and price greater than 4000
* db.items.find({$or:[{rating : {$lt: 3.5}}, {{price:{$gt:4000}}}]})//ratings less than 3.5 or price greater than 4000
* db.items.find({rating : {$lt: 3.5}}, {rating : 1})//ratings less than 3.5 and only display rating in result
* db.items.find({rating : {$lt: 3.5}}, {rating : 1, qty : 1})//ratings less than 3.5 and return rating and quantity in result

Deleting an item:

db.items.deleteOne({price:22000}) //will delete first entry in case of multiple documents with price equal to 22000.

db.items.deleteMany({price:22000}) //will delete all entries with price equal to 22000.

Updating data:

db.items.updateOne({name : “aaa”},{$set : {price : 10000}}) //will update price to 10000 first entry in case of multiple documents with name equal to aaa.

db.items.updateMany({name : “aaa”},{$set : {price : 10000 , rating : 1}}) //will update price to 10000 and rating to 1 for first entry in case of multiple documents with name equal to aaa.

To connect mongodb in a node.js project

1. install mongoose in your project. It is an