How to Install MongoDB on Mac | Install MongoDB on macOS (2024)

https://www.youtube.com/watch?v=8gUQL2zlpvI

DEMO LINK:
https://www.awesomescreenshot.com/video/30642594?key=2c60b15ad1555a5b3dcbe1b36ee89d56

Method 2: Using Node Version Manager (NVM)

NVM is a version manager for Node.js, which allows you to install and switch between multiple versions of Node.js effortlessly.

1. Install NVM:
   Run the following command to install NVM:
2. bash

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.1/install.sh | bash
```

3. Alternatively, visit the NVM GitHub page to check for the latest installation script if there's a newer version than `v0.39.1`.
4. Load NVM:
   After installation, add NVM to your shell profile. Typically, this involves adding the following lines to your `.zshrc` or `.bash_profile`:
5. bash

```
export NVM_DIR="$([ -z "${XDG_CONFIG_HOME-}" ] && printf %s "${HOME}/.nvm" || printf %s "${XDG_CONFIG_HOME}/nvm")"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
```

6. For Zsh, update `.zshrc`:
7. bash

```
nano ~/.zshrc
```

8. Add the lines to the end of the file, save, and reload the terminal, or source the file directly:
9. bash

```
source ~/.zshrc
```

10.
11. Install Node.js:
    Use NVM to install Node.js:

12. bash

nvm install node

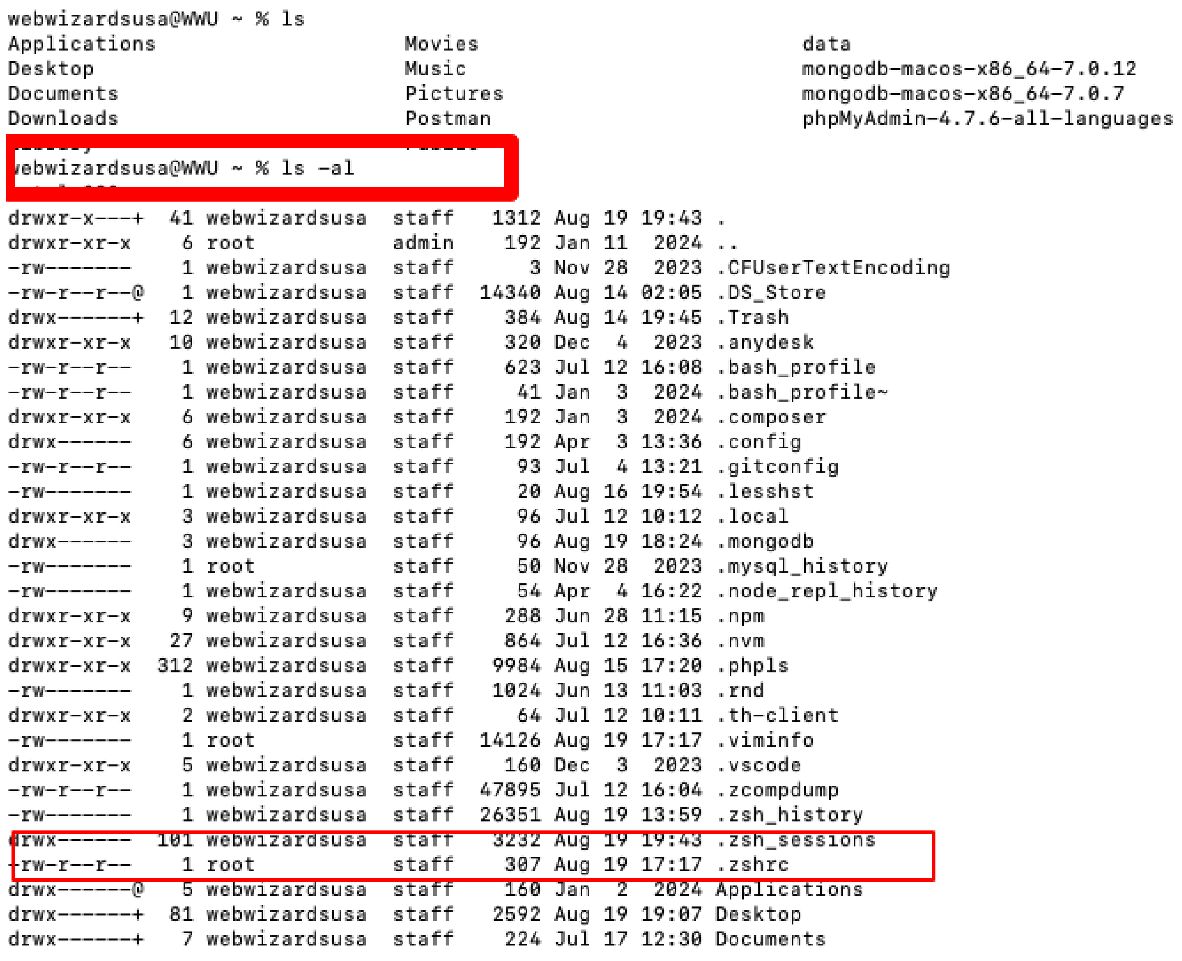
13. This will install the latest version of Node.js. If you need a specific version, replace `node` with, for example, `14.17.0`.
14. Verify Installation:
    Again, verify by checking the versions:
15. bash

node -v

npm -v

16.

Ls -al

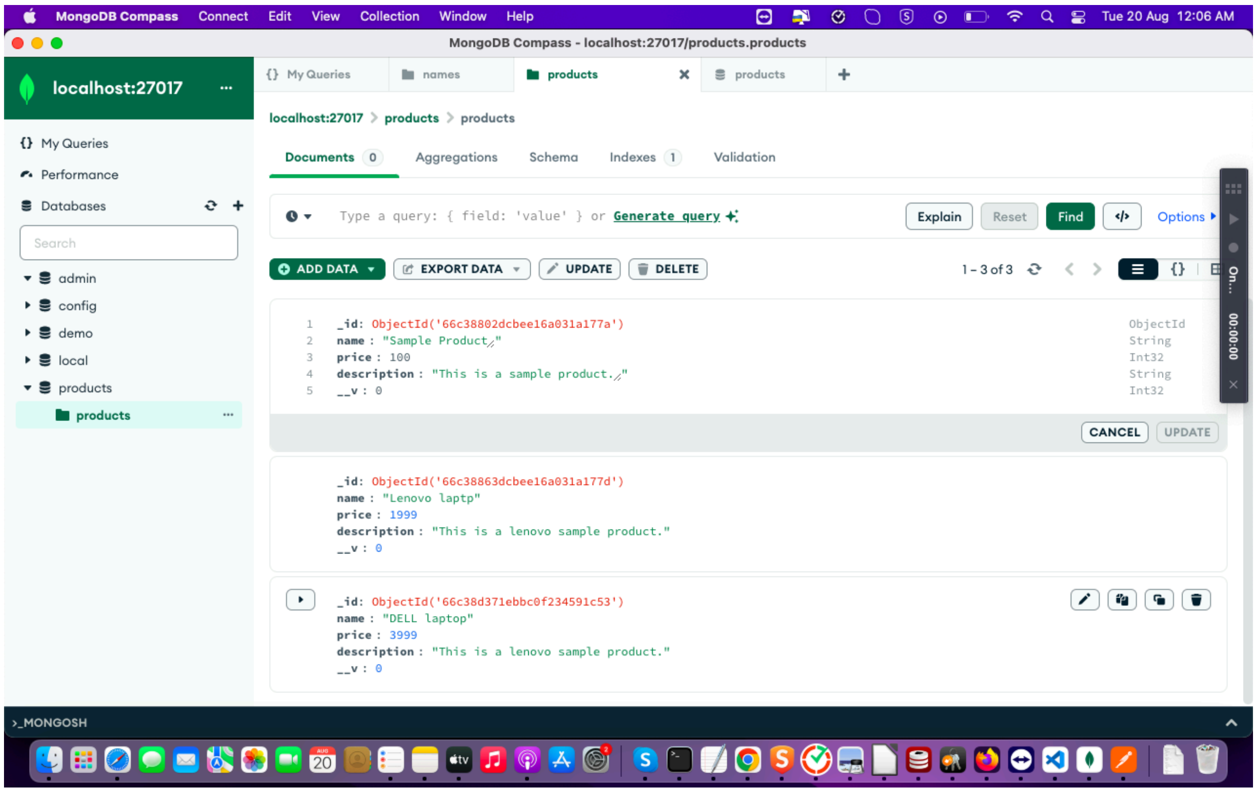


```
webwizardsusa@WWU ~ % ls
Applications                    Movies                          data
Desktop                         Music                           mongodb-macos-x86_64-7.0.12
Documents                       Pictures                        mongodb-macos-x86_64-7.0.7
Downloads                       Postman                         phpMyAdmin-4.7.6-all-languages
webwizardsusa@WWU ~ % ls -al
drwxr-x---+  41 webwizardsusa  staff   1312 Aug 19 19:43 .
drwxr-xr-x    6 root           admin    192 Jan 11  2024 ..
-rw-------    1 webwizardsusa  staff      3 Nov 28  2023 .CFUserTextEncoding
-rw-r--r--@   1 webwizardsusa  staff  14340 Aug 14 02:05 .DS_Store
drwx------+  12 webwizardsusa  staff    384 Aug 14 19:45 .Trash
drwxr-xr-x   10 webwizardsusa  staff    320 Dec  4  2023 .anydesk
-rw-r--r--    1 webwizardsusa  staff    623 Jul 12 16:08 .bash_profile
-rw-r--r--    1 webwizardsusa  staff     41 Jan  3  2024 .bash_profile~
drwxr-xr-x    6 webwizardsusa  staff    192 Jan  3  2024 .composer
drwx------    6 webwizardsusa  staff    192 Apr  3 13:36 .config
-rw-r--r--    1 webwizardsusa  staff     93 Jul  4 13:21 .gitconfig
-rw-------    1 webwizardsusa  staff     20 Aug 16 19:54 .lesshst
drwxr-xr-x    3 webwizardsusa  staff     96 Jul 12 10:12 .local
drwx------    3 webwizardsusa  staff     96 Aug 19 18:24 .mongodb
-rw-------    1 root           staff     50 Nov 28  2023 .mysql_history
-rw-------    1 webwizardsusa  staff     54 Apr  4 16:22 .node_repl_history
drwxr-xr-x    9 webwizardsusa  staff    288 Jun 28 11:15 .npm
drwxr-xr-x   27 webwizardsusa  staff    864 Jul 12 16:36 .nvm
drwxr-xr-x  312 webwizardsusa  staff   9984 Aug 15 17:20 .phpls
-rw-------    1 webwizardsusa  staff   1024 Jun 13 11:03 .rnd
drwxr-xr-x    2 webwizardsusa  staff     64 Jul 12 10:11 .th-client
-rw-------    1 root           staff  14126 Aug 19 17:17 .viminfo
drwxr-xr-x    5 webwizardsusa  staff    160 Dec  3  2023 .vscode
-rw-r--r--    1 webwizardsusa  staff  47895 Jul 12 16:04 .zcompdump
-rw-------    1 webwizardsusa  staff  26351 Aug 19 13:59 .zsh_history
drwx------  101 webwizardsusa  staff   3232 Aug 19 19:43 .zsh_sessions
-rw-r--r--    1 root           staff    307 Aug 19 17:17 .zshrc
drwx------@   5 webwizardsusa  staff    160 Jan  2  2024 Applications
drwx------+  81 webwizardsusa  staff   2592 Aug 19 19:07 Desktop
drwx------+   7 webwizardsusa  staff    224 Jul 17 12:30 Documents
```

/Users/webwizardsusa/mongodb-macos-x86_64-7.0.12/bin

Sudo mongod --dbpath=/Users/webwizardsusa/data/db

mongodb://localhost:27017



```
_id: ObjectId('66c38802dcbee16a031a177a')      ObjectId
name : "Sample Product"                         String
price : 100                                     Int32
description : "This is a sample product."       String
__v : 0                                         Int32
```

```
_id: ObjectId('66c38863dcbee16a031a177d')
name : "Lenovo laptp"
price : 1999
description : "This is a lenovo sample product."
__v : 0
```

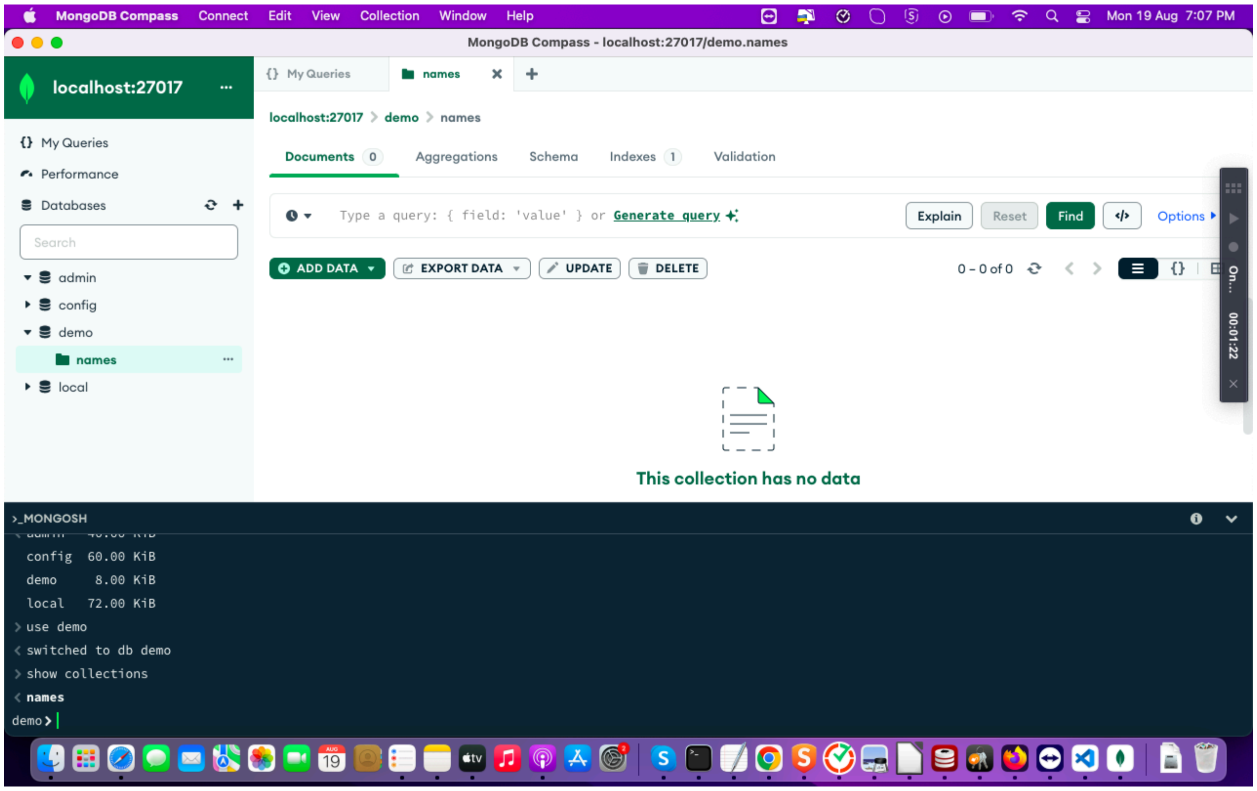
```
_id: ObjectId('66c38d371ebbc0f234591c53')
name : "DELL laptop"
price : 3999
description : "This is a lenovo sample product."
__v : 0
```

```
show dbs;
```

```
admin    40.00 KiB

config   60.00 KiB

demo      8.00 KiB
```

```
local    72.00 KiB

use demo

switched to db demo

show collections

names
```

```
demo
```



This collection has no data

```
  config  60.00 KiB
  demo     8.00 KiB
  local   72.00 KiB
> use demo
< switched to db demo
> show collections
< names
demo>
```

Step 1: Initial Setup

1. Initialize Project
2. bash

mkdir express-product-crud
cd express-product-crud
npm init -y

3.
4. Install Dependencies
5. bash

npm install express mongoose multer body-parser express-validator

6.
   - `express`: Web framework.
   - `mongoose`: MongoDB ORM for defining schemas and interacting with the database.
   - `multer`: Middleware for handling file uploads.
   - `body-parser`: Middleware to parse request bodies.
   - `express-validator`: Middleware for validating request data.

Step 2: Create the Database Schema

Create a new file `models/Product.js`:

```javascript
const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
    minlength: 3,
    maxlength: 50
  },
  price: {
    type: Number,
    required: true,
    min: 0
  },
  description: {
    type: String,
    maxlength: 500
  },
  imageUrl: {
    type: String
  }
});

module.exports = mongoose.model('Product', productSchema);
```

Step 3: Set Up Express Server and Routes

Create a server file `server.js`:

```javascript
const express = require('express');
const mongoose = require('mongoose');
const Product = require('./models/Product');
const multer = require('multer');
const bodyParser = require('body-parser');
const { body, validationResult } = require('express-validator');

const app = express();
const upload = multer({ dest: 'uploads/' });

app.use(bodyParser.json());

mongoose.connect('mongodb://localhost:27017/products', {
  useNewUrlParser: true,
  useUnifiedTopology: true
});

// CREATE a product
app.post('/products', [
  body('name').isString().isLength({ min: 3, max: 50 }),
  body('price').isFloat({ min: 0 }),
  body('description').optional().isString().isLength({ max: 500 })
], async (req, res) => {
  const errors = validationResult(req);
  if (!errors.isEmpty()) {
    return res.status(400).json({ errors: errors.array() });
  }

  const product = new Product(req.body);
  await product.save();
  res.status(201).json(product);
});

// GET all products
app.get('/products', async (req, res) => {
  const products = await Product.find();
  res.json(products);
});

// UPDATE a product by id
app.put('/products/:id', [
  body('name').optional().isString().isLength({ min: 3, max: 50 }),
  body('price').optional().isFloat({ min: 0 }),
  body('description').optional().isString().isLength({ max: 500 })
], async (req, res) => {
  const errors = validationResult(req);
  if (!errors.isEmpty()) {
    return res.status(400).json({ errors: errors.array() });
  }
  const product = await Product.findByIdAndUpdate(req.params.id, req.body, { new: true });
  if (!product) {
    return res.status(404).send('Product not found');
  }
  res.json(product);
});

// DELETE a product by id
app.delete('/products/:id', async (req, res) => {
  const product = await Product.findByIdAndDelete(req.params.id);
  if (!product) {
    return res.status(404).send('Product not found');
  }
  res.status(204).send();
});

// FILE UPLOAD to a product
app.post('/products/:id/upload', upload.single('image'), async (req, res) => {
  const product = await Product.findById(req.params.id);
  if (!product) {
    return res.status(404).send('Product not found');
  }
  product.imageUrl = `/uploads/${req.file.filename}`;
  await product.save();
  res.json(product);
});

// Start the server
app.listen(3000, () => {
  console.log('Server running on port 3000');
});
```

Explanation

- CRUD Operations: Implemented via standard HTTP methods (GET, POST, PUT, DELETE).
- Validation: Utilizes `express-validator` to ensure that incoming data meets the schema requirements.
- File Uploads: Uses `multer` to handle file uploads, storing them in an `uploads/` directory and associating the file path with a product.

Running the Project

1. Start MongoDB if it's not already running:
2. bash

```
mongod
```

3.
4. Run the Express Server:
5. bash

```
node server.js
```

6.

With these steps, you have a basic Express.js project to handle CRUD operations, file uploads, and data validation for a Product resource. You can expand this example further by adding additional features or integrating with front-end applications as needed.

##### end ####
If you see the error `zsh: command not found: mongod`, it means that MongoDB is not installed, or its binaries are not properly set in your system PATH. Here's how you can install MongoDB on a macOS system using Homebrew, which is a package manager for macOS:

Step-by-Step Installation of MongoDB using Homebrew

1. Install Homebrew:
   If you don't have Homebrew installed, open your terminal and run the following command to install it:
2. bash

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

3.
4. Add MongoDB Tap:
   MongoDB is no longer part of the default Homebrew repository, so you need to add the `mongodb/brew` tap first:
5. bash

```
brew tap mongodb/brew
```

6.

7. Install MongoDB:
   Use Homebrew to install MongoDB. This will install MongoDB Server, MongoDB Shell, and some other utilities:
8. bash

```
brew install mongodb-community@5.0
```

9. (Replace `5.0` with the desired version of MongoDB, if different)
10. Start MongoDB as a Service:
    Once installed, you can start the MongoDB server using Homebrew's service management commands:
11. bash

```
brew services start mongodb/brew/mongodb-community
```

12. Alternatively, if you prefer running MongoDB manually, you can use:
13. bash

```
mongod --config /usr/local/etc/mongod.conf
```

14.
15. Verify Installation:
    After starting the MongoDB service, verify the installation by connecting to the MongoDB shell:
16. bash

```
mongo
```

17. If you see the MongoDB shell prompt (`>`), the installation succeeded and MongoDB is running.
18. Set MongoDB to Start at Boot (Optional):
    You can set MongoDB to start automatically when your macOS boots up. This is generally a good idea if you plan to use MongoDB regularly:
19. bash

```
brew services start mongodb/brew/mongodb-community
```

20.

Troubleshooting Tips

- PATH Issue: If you receive a `command not found` error even after installation, confirm that the MongoDB binaries are in your PATH. You can add `/usr/local/bin` to your PATH in your shell configuration file (`.zshrc`, `.bash_profile`, etc.).

- Running Services: Ensure that no other service conflicts with the ports MongoDB uses (default port 27017).
- Logs: If MongoDB fails to start, check the logs at `/usr/local/var/log/mongodb/mongo.log` for any errors.

Following these steps will install MongoDB on your macOS system and make it ready to run with Homebrew's service management. Remember to keep MongoDB updated periodically using Homebrew commands (`brew update` and `brew upgrade mongodb-community`).

####### “`mongod` can't be opened because Apple cannot check it for malicious software” ##

The message “`mongod` can't be opened because Apple cannot check it for malicious software” is a security feature of macOS that is part of its Gatekeeper system. This system helps protect your Mac from software that could potentially harm your device by restricting apps that are not from the App Store or are not signed with a recognized developer ID.

Here's how you can work around this issue and open `mongod`:

Method 1: Use System Preferences to Allow the App

1. Open System Preferences:
   Go to `System Preferences` > `Security & Privacy`.
2. Navigate to the General Tab:
   Ensure you're on the "General" tab.
3. Allow the App:
   - You should see a message at the bottom similar to "mongod was blocked from use because it is not from an identified developer."
   - Click on the "Open Anyway" button next to this message.
4. Re-launch `mongod`:
   Try running `mongod` again:
5. bash

mongod --config /usr/local/etc/mongod.conf

6.

Method 2: Use the Terminal to Bypass Gatekeeper Temporarily

If the above method doesn't work or if you want to use Terminal entirely, you can use the command line to open MongoDB:

1. Run the MongoDB Command:
   Use the following command to run `mongod`, acknowledging all security prompts:
2. bash

sudo xattr -rd com.apple.quarantine /usr/local/bin/mongod

3. This command removes the quarantine attribute from the `mongod` binary, which is the attribute that tells macOS the app was downloaded and needs checking.

Method 3: Use the Context Menu (Quick Alternative)

1. Locate MongoDB in Finder:
   Navigate to `/usr/local/bin` in Finder, or wherever `mongod` is installed. This path might vary depending on how MongoDB was installed.
2. Open the Application via Right-Click:
   - Right-click or Control-click `mongod`.
   - Select `Open`.
   - Confirm again when prompted.

This action should bypass the Gatekeeper warning for this instance.

Important Notes

- Background Information: This Gatekeeper warning often appears when software isn't notarized by recognized developers, or when it's a command-line tool without a graphical interface.
- Security Considerations: Always ensure that you're downloading software from trustworthy sources to avoid security risks.
- Repeat Understanding: If the app is updated or reinstalled, you might have to repeat these steps.

By following these steps, you can run `mongod` on your macOS device even if it's blocked by Gatekeeper.