Using mongoose to communicate between express and mongo

Database communications

Communicating with an SQL database a user can either use native SQL commands or an intermediary programme which provides an Object Data Model ("ODM") / Object Relational Model ("ORM").

The native approach produces the fastest response from the database.

The model approach allows the app to be developed to work with a range of available databases using the same code.

The ODM used here to address the noSQL database Mongo is called Mongoose.

A mongoose primer is available from Mozilla developer.

Mongoose

Mongoose will be added to the express app to alow it to communicate with the database.

From the exercises preceeding we have developed an express app in folder express_app2.

Return to the express application. Copy the folder express_app2 as express_app3 and change directories to this in a new terminal window.

Continue to dockerize the tutorial

The dockerfile and package.json will be updated to add mongoose into the image.

Change the version of node up to version 12.

```
FROM node:12
ENV DEBUG="myapp:*"

# Create app directory
RUN mkdir -p /usr/src/app
COPY  ./myapp/package.json  /usr/src/app/package.json

WORKDIR /usr/src/app
RUN npm install && mv /usr/src/app/node_modules /node_modules ;\
    npm install -g nodemon

COPY ./myapp /usr/src/app


VOLUME ["/usr/src/app"]
EXPOSE 3000
CMD [ "npm","run","start" ]
```

Edit package.json to ensure that mongoose is added to the dependancies. I have also added body-parser, which will be useful later on.

```
{
  "name": "myapp",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "nodemon -L start.js"
  },
  "dependencies": {
    "cookie-parser": "~1.4.4",
    "debug": "~2.6.9",
    "express": "~4.16.1",
    "http-errors": "~1.6.3",
    "morgan": "~1.9.1",
    "pug": "2.0.0-beta11",
    "nodemon": "1.19.1",
    "mongoose":"^5.7.8",
    "body-parser":"^1.19.0"
  },
  "nodemonConfig": {
    "delay": "1500",
    "verbose": "true"
  }
}
```

Open app.js and add code to setup the mongoose connection immediately below the reference to var app = express();

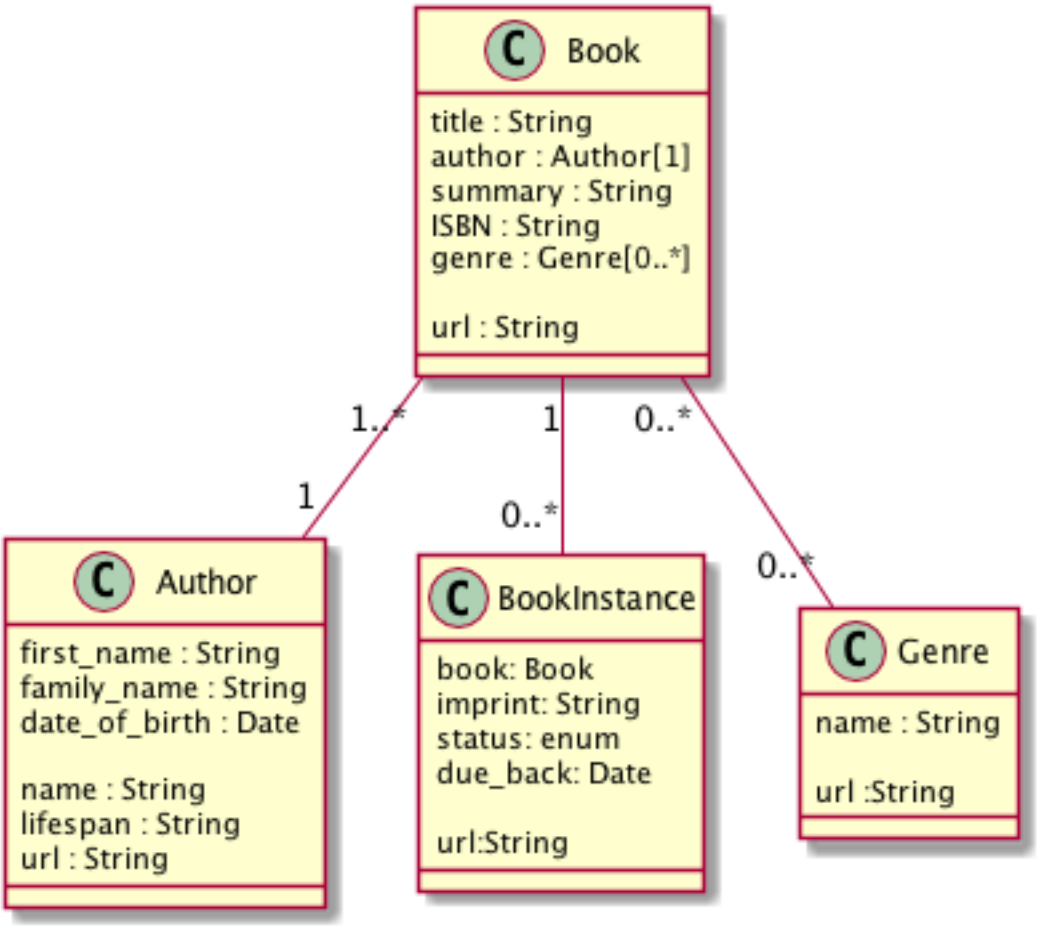
app.js (extract)

```
var app = express();
//Set up mongoose connection
const mongoose = require('mongoose');
const mongoDB = 'mongodb://mongo:27017/myapp';
mongoose.Promise = global.Promise;
mongoose.connect(mongoDB, {
  useNewUrlParser: true,
  useUnifiedTopology: true,
  //auth: {
  //user: "root",
  //password: "example"
  //}
});
const db = mongoose.connection;
db.on('error', console.error.bind(console, 'MongoDB connection error:'));
```

Note that the connection string for mongoDB is the database url many references describe localhost:27017/dbname.

As this example is developed we will launch both the database and the express application from a single docker compose file. In this case docker creates a bridge network between the containers and the connection string format becomes mongodb://databaseContainerName:27017/applicationContainerName. References to localhost will not work. This is an essential point to note as it is make or break to connecting application and database.

To work with an Object Data Model we need to define objects which relate to the database.



database uml

Now create a models folder and in this create each of author.js, book.js. bookinstance.js and genre.js.

express_app3 /myapp /models author.js book.js bookinstance.js genre.js

author.js

```
var mongoose = require('mongoose');

var Schema = mongoose.Schema;
```

```
var AuthorSchema = new Schema(
  {
    first_name: {type: String, required: true, max: 100},
    family_name: {type: String, required: true, max: 100},
    date_of_birth: {type: Date},
    date_of_death: {type: Date},
  }
);

// Virtual for author's full name
AuthorSchema
.virtual('name')
.get(function () {
  return this.family_name + ', ' + this.first_name;
});

// Virtual for author's lifespan
AuthorSchema
.virtual('lifespan')
.get(function () {
  return (this.date_of_death.getYear() -
this.date_of_birth.getYear()).toString();
});

// Virtual for author's URL
AuthorSchema
.virtual('url')
.get(function () {
  return '/catalog/author/' + this._id;
});

//Export model
module.exports = mongoose.model('Author', AuthorSchema);
```

book.js

```
var mongoose = require('mongoose');

var Schema = mongoose.Schema;

var BookSchema = new Schema(
  {
    title: {type: String, required: true},
    author: {type: Schema.Types.ObjectId, ref: 'Author', required: true},
    summary: {type: String, required: true},
    isbn: {type: String, required: true},
    genre: [{type: Schema.Types.ObjectId, ref: 'Genre'}]
  }
);
```

```
// Virtual for book's URL
BookSchema
.virtual('url')
.get(function () {
  return '/catalog/book/' + this._id;
});

//Export model
module.exports = mongoose.model('Book', BookSchema);
```

bookinstance.js

```
var mongoose = require('mongoose');

var Schema = mongoose.Schema;

var BookInstanceSchema = new Schema(
  {
    book: { type: Schema.Types.ObjectId, ref: 'Book', required: true },
//reference to the associated book
    imprint: {type: String, required: true},
    status: {type: String, required: true, enum: ['Available', 'Maintenance',
'Loaned', 'Reserved'], default: 'Maintenance'},
    due_back: {type: Date, default: Date.now}
  }
);

// Virtual for bookinstance's URL
BookInstanceSchema
.virtual('url')
.get(function () {
  return '/catalog/bookinstance/' + this._id;
});

//Export model
module.exports = mongoose.model('BookInstance', BookInstanceSchema);
```

genre.js

```
var mongoose = require('mongoose');

var Schema = mongoose.Schema;

var GenreSchema = new Schema(
  {
    name: {type: String, required: true}
  }
);

// Virtual for genre's URL
```

```
GenreSchema
.virtual('url')
.get(function () {
  return '/catalog/genre/' + this._id;
});

//Export model
module.exports = mongoose.model('Genre', GenreSchema);
```

Now continue to dockerize the next tutorial on Routes and controllers

Need to decide upon the URLs that will be needed to access the library data required.

- catalog/ - index page
- catalog <objects>
  catalog/books
  catalog/authors
  catalog/genres
  catalog/bookinstances
- catalog/<object>/<id>
  catalog/book/5899af...23A
  catalog/author/5749af...25A
  catalog/genre/5823af...2AB
  catalog/bookinstance/4799af...63A
- catalog/<object>/create
  catalog/book/create
  catalog/author/create
  catalog/genre/create
  catalog/bookinstance/create
- catalog/<object>/<id>update
  catalog/book/5899af...23A/update
  catalog/author/5749af...25A/update
  catalog/genre/5823af...2AB/update
  catalog/bookinstance/4799af...63A/update
- catalog/<object>/<id>delete
  catalog/book/5899af...23A/delete
  catalog/author/5749af...25A/delete
  catalog/genre/5823af...2AB/delete
  catalog/bookinstance/4799af...63A/delete

Now need a route for each resource of a particular type.

Create route handler callback functions in the folder controllers.

```
express_app3
  /myapp
    /controllers
       authorController.js
       bookController.js
       bookinstanceController.js
       genreController.js
```

authorController.js

```
var Author = require('../models/author');

// Display list of all Authors.
exports.author_list = function(req, res) {
    res.send('NOT IMPLEMENTED: Author list');
};

// Display detail page for a specific Author.
exports.author_detail = function(req, res) {
    res.send('NOT IMPLEMENTED: Author detail: ' + req.params.id);
};

// Display Author create form on GET.
exports.author_create_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Author create GET');
};

// Handle Author create on POST.
exports.author_create_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Author create POST');
};

// Display Author delete form on GET.
exports.author_delete_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Author delete GET');
};

// Handle Author delete on POST.
exports.author_delete_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Author delete POST');
};

// Display Author update form on GET.
exports.author_update_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Author update GET');
};

// Handle Author update on POST.
exports.author_update_post = function(req, res) {
```

```
    res.send('NOT IMPLEMENTED: Author update POST');
};
```

bookinstanceController.js

```
var BookInstance = require('../models/bookinstance');

// Display list of all BookInstances.
exports.bookinstance_list = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance list');
};

// Display detail page for a specific BookInstance.
exports.bookinstance_detail = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance detail: ' + req.params.id);
};

// Display BookInstance create form on GET.
exports.bookinstance_create_get = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance create GET');
};

// Handle BookInstance create on POST.
exports.bookinstance_create_post = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance create POST');
};

// Display BookInstance delete form on GET.
exports.bookinstance_delete_get = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance delete GET');
};

// Handle BookInstance delete on POST.
exports.bookinstance_delete_post = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance delete POST');
};

// Display BookInstance update form on GET.
exports.bookinstance_update_get = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance update GET');
};

// Handle bookinstance update on POST.
exports.bookinstance_update_post = function(req, res) {
    res.send('NOT IMPLEMENTED: BookInstance update POST');
};
```

genreController.js

```
var Genre = require('../models/genre');

// Display list of all Genre.
exports.genre_list = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre list');
};

// Display detail page for a specific Genre.
exports.genre_detail = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre detail: ' + req.params.id);
};

// Display Genre create form on GET.
exports.genre_create_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre create GET');
};

// Handle Genre create on POST.
exports.genre_create_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre create POST');
};

// Display Genre delete form on GET.
exports.genre_delete_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre delete GET');
};

// Handle Genre delete on POST.
exports.genre_delete_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre delete POST');
};

// Display Genre update form on GET.
exports.genre_update_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre update GET');
};

// Handle Genre update on POST.
exports.genre_update_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Genre update POST');
};
```

bookController.js

Has an extra index() function to display a site welcome page.

```
var Book = require('../models/book');

exports.index = function(req, res) {
    res.send('NOT IMPLEMENTED: Site Home Page');
```

```
};

// Display list of all books.
exports.book_list = function(req, res) {
    res.send('NOT IMPLEMENTED: Book list');
};

// Display detail page for a specific book.
exports.book_detail = function(req, res) {
    res.send('NOT IMPLEMENTED: Book detail: ' + req.params.id);
};

// Display book create form on GET.
exports.book_create_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Book create GET');
};

// Handle book create on POST.
exports.book_create_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Book create POST');
};

// Display book delete form on GET.
exports.book_delete_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Book delete GET');
};

// Handle book delete on POST.
exports.book_delete_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Book delete POST');
};

// Display book update form on GET.
exports.book_update_get = function(req, res) {
    res.send('NOT IMPLEMENTED: Book update GET');
};

// Handle book update on POST.
exports.book_update_post = function(req, res) {
    res.send('NOT IMPLEMENTED: Book update POST');
};
```

We now need routes which will link URLs to the callback handlers.

Create a file catalog.js inside the routes folder which currently contains index and users.

```
var express = require('express');
var router = express.Router();

// Require controller modules.
```

```
var book_controller = require('../controllers/bookController');
var author_controller = require('../controllers/authorController');
var genre_controller = require('../controllers/genreController');
var book_instance_controller =
require('../controllers/bookinstanceController');

/// BOOK ROUTES ///

// GET catalog home page.
router.get('/', book_controller.index);

// GET request for creating a Book. NOTE This must come before routes that
display Book (uses id).
router.get('/book/create', book_controller.book_create_get);

// POST request for creating Book.
router.post('/book/create', book_controller.book_create_post);

// GET request to delete Book.
router.get('/book/:id/delete', book_controller.book_delete_get);

// POST request to delete Book.
router.post('/book/:id/delete', book_controller.book_delete_post);

// GET request to update Book.
router.get('/book/:id/update', book_controller.book_update_get);

// POST request to update Book.
router.post('/book/:id/update', book_controller.book_update_post);

// GET request for one Book.
router.get('/book/:id', book_controller.book_detail);

// GET request for list of all Book items.
router.get('/books', book_controller.book_list);

/// AUTHOR ROUTES ///

// GET request for creating Author. NOTE This must come before route for id
(i.e. display author).
router.get('/author/create', author_controller.author_create_get);

// POST request for creating Author.
router.post('/author/create', author_controller.author_create_post);

// GET request to delete Author.
router.get('/author/:id/delete', author_controller.author_delete_get);
```

```
// POST request to delete Author.
router.post('/author/:id/delete', author_controller.author_delete_post);

// GET request to update Author.
router.get('/author/:id/update', author_controller.author_update_get);

// POST request to update Author.
router.post('/author/:id/update', author_controller.author_update_post);

// GET request for one Author.
router.get('/author/:id', author_controller.author_detail);

// GET request for list of all Authors.
router.get('/authors', author_controller.author_list);

/// GENRE ROUTES ///

// GET request for creating a Genre. NOTE This must come before route that
displays Genre (uses id).
router.get('/genre/create', genre_controller.genre_create_get);

//POST request for creating Genre.
router.post('/genre/create', genre_controller.genre_create_post);

// GET request to delete Genre.
router.get('/genre/:id/delete', genre_controller.genre_delete_get);

// POST request to delete Genre.
router.post('/genre/:id/delete', genre_controller.genre_delete_post);

// GET request to update Genre.
router.get('/genre/:id/update', genre_controller.genre_update_get);

// POST request to update Genre.
router.post('/genre/:id/update', genre_controller.genre_update_post);

// GET request for one Genre.
router.get('/genre/:id', genre_controller.genre_detail);

// GET request for list of all Genre.
router.get('/genres', genre_controller.genre_list);

/// BOOKINSTANCE ROUTES ///

// GET request for creating a BookInstance. NOTE This must come before route
that displays BookInstance (uses id).
router.get('/bookinstance/create',
book_instance_controller.bookinstance_create_get);
```

```
// POST request for creating BookInstance.
router.post('/bookinstance/create',
book_instance_controller.bookinstance_create_post);

// GET request to delete BookInstance.
router.get('/bookinstance/:id/delete',
book_instance_controller.bookinstance_delete_get);

// POST request to delete BookInstance.
router.post('/bookinstance/:id/delete',
book_instance_controller.bookinstance_delete_post);

// GET request to update BookInstance.
router.get('/bookinstance/:id/update',
book_instance_controller.bookinstance_update_get);

// POST request to update BookInstance.
router.post('/bookinstance/:id/update',
book_instance_controller.bookinstance_update_post);

// GET request for one BookInstance.
router.get('/bookinstance/:id',
book_instance_controller.bookinstance_detail);

// GET request for list of all BookInstance.
router.get('/bookinstances', book_instance_controller.bookinstance_list);

module.exports = router;
```

Edit the existing index.js to redirect the index page to the index created at path /catalog.

```
// GET home page.
router.get('/', function(req, res) {
  res.redirect('/catalog');
});
```

The use of the redirect() method will send HTTP status code “302 Found”.

The last step in setting up routes is to add them into app.js below existing routes.

```
var indexRouter = require('./routes/index');
var usersRouter = require('./routes/users');
var catalogRouter = require('./routes/catalog');  //Import routes for
"catalog" area of site
```

and also in app.js

```
app.use('/', indexRouter);
app.use('/users', usersRouter);
```

```
app.use('/catalog', catalogRouter);  // Add catalog routes to middleware
chain.
```

These routes don't yet access the database but should print messages to show they are working.

The complete listing of app.js at this point is

```
var createError = require('http-errors');
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');

var indexRouter = require('./routes/index');
var usersRouter = require('./routes/users');
var catalogRouter = require('./routes/catalog');  //Import routes for
"catalog" area of site

var app = express();
//Set up mongoose connection
const mongoose = require('mongoose');
const mongoDB = 'mongodb://mongo:27017/myapp';
mongoose.Promise = global.Promise;
mongoose.connect(mongoDB, {
  useNewUrlParser: true,
  useUnifiedTopology: true,
  //auth: {
  //user: "root",
  //password: "example"
  //}
});
const db = mongoose.connection;
db.on('error', console.error.bind(console, 'MongoDB connection error:'));

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'pug');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', indexRouter);
app.use('/users', usersRouter);
app.use('/catalog', catalogRouter);  // Add catalog routes to middleware
chain.
```

```
// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;
```

Launching containers using Docker Compose

As we have worked so far we have working folders, mongo1:

```
mongo1\
  mongo-init.js
  stack.yml
```

mongo-init.js contains the data for the database to be loaded at the time of docker compose stack.yml up.

We have a docker image named “mongo” of the mongo database which was pulled from docker hub.

We also have a docker image named “mongo-express” which contains the administrator interface for the mongo database. This will not be part of the final application, but it is useful for inspecting the database to ensure that the application is working with the database properly. This interface also lets an administrator add and delete users and database from the mongo database.

In the folder express_app3 we have the files required for an express application routing queries about the library to response messages.

You can also see here where you are about to copy stack.yml and mongo-init.js (but don’t copy these yet).

```
express_app3
  Dockerfile
  stack.yml
  mongo-init.js
  /myapp
    app.js
```

```
start.js
package.json
/bin
  www
/controllers
   authorController.js
   bookController.js
   bookinstanceController.js
   genreController.js
/models
   author.js
   book.js
   bookinstance.js
   genre.js
/public
  /images
  /javascripts
  /stylesheets
/routes
  catalog.js
  index.js
  users.js
/views
  error.pug
  index.pug
  layout.pug
```

We also have and old verion of our image express-development which was based on Node 10. We will need to delete this.

There may also be a number of stopped containers from code development which are no longer needed.

Look for the old dt/express-development image, find its reference id and delete it.

docker image ls

docker rm imageID -f

Tidy up and remove any other unwanted images.

Remove exited containers

docker rm ls -f 'status=exited' | xargs docker rm

Generally tidy up your docker assets.

Within express_app3

First need to build dt/express-development image again so that the image has mongoose loaded.

docker build -t dt/express-development .

Terminal output includes some warnings which are noted but not actioned.

```
Sending build context to Docker daemon  41.98kB
Step 1/10 : FROM node:12
12: Pulling from library/node
7919f5b7d602: Already exists
0e107167dcc5: Already exists
66a456bba435: Already exists
5435318a0426: Already exists
8494dd328465: Already exists
3b01939c6506: Already exists
53e99ce7f062: Pull complete
7e6608095b28: Pull complete
d75bbd3f7c4b: Pull complete
Digest:
sha256:6e2db75c0a1e19ed760996957aef507f5abe1260ec412e8901855ac4a17a7ada
Status: Downloaded newer image for node:12
 ---> e4f1e16b3633
Step 2/10 : ENV DEBUG="myapp:*"
 ---> Running in f262e648dc6f
Removing intermediate container f262e648dc6f
 ---> ee3c9be30ce5
Step 3/10 : RUN mkdir -p /usr/src/app
 ---> Running in 4930bb64117d
Removing intermediate container 4930bb64117d
 ---> be1cfb7ab6c8
Step 4/10 : COPY  ./myapp/package.json  /usr/src/app/package.json
 ---> d3b4b72b0201
Step 5/10 : WORKDIR /usr/src/app
 ---> Running in c2064bdf157d
Removing intermediate container c2064bdf157d
 ---> 3a1c6e7785ca
Step 6/10 : RUN npm install && mv /usr/src/app/node_modules /node_modules ;
npm install -g nodemon
 ---> Running in 3ec9f3ec9777
npm WARN deprecated chokidar@2.1.8: Chokidar 2 will break on node v14+.
Upgrade to chokidar 3 with 15x less dependencies.
npm WARN deprecated fsevents@1.2.13: fsevents 1 will break on node v14+ and
could be using insecure binaries. Upgrade to fsevents 2.
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-
url#deprecated
npm WARN deprecated urix@0.1.0: Please see
https://github.com/lydell/urix#deprecated
npm WARN deprecated core-js@2.6.12: core-js@<3 is no longer maintained and
not recommended for usage due to the number of issues. Please, upgrade your
dependencies to the actual version of core-js@3.

> core-js@2.6.12 postinstall /usr/src/app/node_modules/core-js
```

```
> node -e "try{require('./postinstall')}catch(e){}"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for
polyfilling JavaScript standard library!

The project needs your help! Please consider supporting of core-js on Open
Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a
good job -)


> nodemon@1.19.1 postinstall /usr/src/app/node_modules/nodemon
> node bin/postinstall || exit 0

Love nodemon? You can now support the project via the open collective:
 > https://opencollective.com/nodemon/donate

npm notice created a lockfile as package-lock.json. You should commit this
file.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@^1.2.7
(node_modules/chokidar/node_modules/fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for
fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current:
{"os":"linux","arch":"x64"})

added 373 packages from 341 contributors and audited 375 packages in 19.648s

6 packages are looking for funding
  run `npm fund` for details

found 1 low severity vulnerability
  run `npm audit fix` to fix them, or `npm audit` for details
/usr/local/bin/nodemon -> /usr/local/lib/node_modules/nodemon/bin/nodemon.js

> nodemon@2.0.6 postinstall /usr/local/lib/node_modules/nodemon
> node bin/postinstall || exit 0

Love nodemon? You can now support the project via the open collective:
 > https://opencollective.com/nodemon/donate

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@~2.1.2
(node_modules/nodemon/node_modules/chokidar/node_modules/fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for
fsevents@2.1.3: wanted {"os":"darwin","arch":"any"} (current:
{"os":"linux","arch":"x64"})
```

```
+ nodemon@2.0.6
added 119 packages from 53 contributors in 10.283s
Removing intermediate container 3ec9f3ec9777
 ---> 86a3c990bb93
Step 7/10 : COPY ./myapp /usr/src/app
 ---> b63368470e5e
Step 8/10 : VOLUME ["/usr/src/app"]
 ---> Running in b3375f0bc3f6
Removing intermediate container b3375f0bc3f6
 ---> f9e2e6b6d580
Step 9/10 : EXPOSE 3000
 ---> Running in 6b8fa5e274f7
Removing intermediate container 6b8fa5e274f7
 ---> 88ba64d27407
Step 10/10 : CMD [ "npm","run","start" ]
 ---> Running in 1812b8032550
Removing intermediate container 1812b8032550
 ---> f9569ba896e0
Successfully built f9569ba896e0
Successfully tagged dt/express-development:latest
SECURITY WARNING: You are building a Docker image from Windows against a non-
Windows Docker host. All files and directories added to build context will
have '-rwxr-xr-x' permissions. It is recommended to double check and reset
permissions for sensitive files and directories.
```

Now copy stack.yml and mongo-init.js from the mongo1 folder into express_app3 folder.

Edit stack.yml so that it will start this app, the mongo database and the mongo admin site.

Stack.yml

```yaml
# Use root/example as user/password credentials
# Try to start up myapp 3 as well  minimum version 3.2
version: '3.8'

volumes:
  dbdata:
  dbconfig:

services:
  mongo:
    image: mongo
    container_name: mongo
    restart: always
    environment:
      MONGO_INITDB_ROOT_USERNAME: root
      MONGO_INITDB_ROOT_PASSWORD: example
      MONGO_INITDB_DATABASE: local_library
    volumes:
    - type: volume
```

```
      source: dbdata
      target: /data/db
    - type: volume
      source: dbconfig
      target: /data/configdb
    - ./mongo-init.js:/docker-entrypoint-initdb.d/mongo-init.js:ro

    ports:
    - 27017:27017

  myapp3:
    image: dt/express-development
    container_name: myapp
    restart: always
    ports:
    - 3000:3000
    environment:
      ME_CONFIG_MONGODB_ADMINUSERNAME: root
      ME_CONFIG_MONGODB_ADMINPASSWORD: example
    volumes:
    - type: bind
      source: ./myapp
      target: /usr/src/app
    links:
    - mongo

  mongo-express:
    image: mongo-express
    restart: always
    ports:
    - 8081:8081
    environment:
      ME_CONFIG_MONGODB_ADMINUSERNAME: root
      ME_CONFIG_MONGODB_ADMINPASSWORD: example
```

I have set the version of the docker compose format to 3.8. The syntax employed here requires at least version 3.2. You can tell whether this is all compatible by looking at docker version.

docker version

```
Client: Docker Engine - Community
 Cloud integration: 1.0.2
 Version:           20.10.0-rc1
 API version:       1.41
 Go version:        go1.13.15
 Git commit:        5cc2396
 Built:             Tue Nov 17 22:49:28 2020
 OS/Arch:           windows/amd64
 Context:           default
```

```
 Experimental:      true

Server: Docker Engine - Community
 Engine:
  Version:          20.10.0-rc1
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.13.15
  Git commit:       131bf7e
  Built:            Tue Nov 17 22:52:57 2020
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          v1.4.1
  GitCommit:        c623d1b36f09f8ef6536a057bd658b3aa8632828
 runc:
  Version:          1.0.0-rc92
  GitCommit:        ff819c7e9184c13b7c2607fe6c30ae19403a7aff
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

My machine has docker version 20. On the docker website the docker-compose format is discussed. The version 3.8 requires docker version 19.03.0 or greater. Look through this reference to review some of the syntax for the docker compose file.

I have started the mongo service from the image mongo and will run it in a container named mongo

I have also added two named volumes dbdata and dbconfig which will persist the contents and configuration of the database. These point to /data/db and /data/configdb which are the default paths used by mongodb to store this information.

Using type:volume means that local storage will be in the docker folder which you generally dont see in file explorer. The volumes can be inspected using docker volume commands. Further details of volumes.

The database volume line

```
- ./mongo-init.js:/docker-entrypoint-initdb.d/mongo-init.js:ro
```

is used to initialise the database on startup. The database exposes its default ports 27017.

The service myapp3 is based on the image dt/express-development and opens in a container name “myapp”

This is needed to form the mongo db connection string.

```
const mongoDB = 'mongodb://mongo:27017/myapp';
```

The app can be viewed in a browser at port 3000.

The volume of type bind means that the host directory ./myapp will be bound to the container directory /usr/src/app The full stop in ./myapp indicates a relative address based on the location of the yaml file.

The mongo-express service is also started for test and debug purposes and is viewed in browser at port 8081.

At the moment there is no security applied to the connection between myapp and mongo. This section is commented out in app.js for later use.

In terminal from express_app3 folder issue command:

docker-compose -f stack.yml up

If you see an error:

```
ERROR: yaml.scanner.ScannerError: mapping values are not allowed here
```

this reflects an layout or spelling error in stack.yml. Pay particular attention to the indentation as this is essential and not decorative.

Terminal output:

```
Creating network "express_app3_default" with the default driver
Creating volume "express_app3_dbdata" with default driver
Creating volume "express_app3_dbconfig" with default driver
Creating mongo                          ... done
Creating express_app3_mongo-express_1 ... done
Creating myapp                          ... done
Attaching to express_app3_mongo-express_1, mongo, myapp
mongo-express_1  | Waiting for mongo:27017...
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Tue Dec  1 03:07:10 UTC 2020 retrying to connect to
mongo:27017 (2/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Tue Dec  1 03:07:11 UTC 2020 retrying to connect to
mongo:27017 (3/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Tue Dec  1 03:07:12 UTC 2020 retrying to connect to
mongo:27017 (4/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Tue Dec  1 03:07:13 UTC 2020 retrying to connect to
mongo:27017 (5/5)
```

```
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo            | about to fork child process, waiting until server is ready
for connections.
mongo            | forked process: 27
mongo            |
mongo            | {"t":{"$date":"2020-12-01T03:07:16.143+00:00"},"s":"I",
"c":"CONTROL",  "id":20698,   "ctx":"main","msg":"***** SERVER RESTARTED
*****"}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.317+00:00"},"s":"I",
"c":"CONTROL",  "id":23285,   "ctx":"main","msg":"Automatically disabling TLS
1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.631+00:00"},"s":"W",
"c":"ASIO",     "id":22601,   "ctx":"main","msg":"No TransportLayer
configured during NetworkInterface startup"}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.794+00:00"},"s":"I",
"c":"NETWORK",  "id":4648601, "ctx":"main","msg":"Implicit TCP FastOpen
unavailable. If TCP FastOpen is required, set tcpFastOpenServer,
tcpFastOpenClient, and tcpFastOpenQueueSize."}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.811+00:00"},"s":"I",
"c":"STORAGE",  "id":4615611, "ctx":"initandlisten","msg":"MongoDB
starting","attr":{"pid":27,"port":27017,"dbPath":"/data/db","architecture":"6
4-bit","host":"82bed784144a"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.812+00:00"},"s":"I",
"c":"CONTROL",  "id":23403,   "ctx":"initandlisten","msg":"Build
Info","attr":{"buildInfo":{"version":"4.4.2","gitVersion":"15e73dc5738d2278b6
88f8929aee605fe4279b0e","openSSLVersion":"OpenSSL 1.1.1  11 Sep
2018","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu180
4","distarch":"x86_64","target_arch":"x86_64"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.814+00:00"},"s":"I",
"c":"CONTROL",  "id":51765,   "ctx":"initandlisten","msg":"Operating
System","attr":{"os":{"name":"Ubuntu","version":"18.04"}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:16.814+00:00"},"s":"I",
"c":"CONTROL",  "id":21951,   "ctx":"initandlisten","msg":"Options set by
command
line","attr":{"options":{"net":{"bindIp":"127.0.0.1","port":27017,"tls":{"mod
e":"disabled"}},"processManagement":{"fork":true,"pidFilePath":"/tmp/docker-
entrypoint-temp-
mongod.pid"},"systemLog":{"destination":"file","logAppend":true,"path":"/proc
/1/fd/1"}}}}
myapp            |
myapp            | > myapp@0.0.0 start /usr/src/app
myapp            | > nodemon -L start.js
myapp            |
mongo            | {"t":{"$date":"2020-12-01T03:07:17.698+00:00"},"s":"I",
"c":"STORAGE",  "id":22297,   "ctx":"initandlisten","msg":"Using the XFS
filesystem is strongly recommended with the WiredTiger storage engine. See
http://dochub.mongodb.org/core/prodnotes-
filesystem","tags":["startupWarnings"]}
```

```
mongo            | {"t":{"$date":"2020-12-01T03:07:17.700+00:00"},"s":"I",
"c":"STORAGE",  "id":22315,   "ctx":"initandlisten","msg":"Opening
WiredTiger","attr":{"config":"create,cache_size=1823M,session_max=33000,evict
ion=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(en
abled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_i
dle_time=100000,close_scan_interval=10,close_handle_minimum=250),statistics_l
og=(wait=0),verbose=[recovery_progress,checkpoint_progress,compact_progress],
"}}
myapp            | [nodemon] 2.0.6
myapp            | [nodemon] reading config ./package.json
myapp            | [nodemon] to restart at any time, enter `rs`
myapp            | [nodemon] or send SIGHUP to 20 to restart
myapp            | [nodemon] watching path(s): *.*
myapp            | [nodemon] watching extensions: js,mjs,json
myapp            | [nodemon] starting `node start.js`
mongo            | {"t":{"$date":"2020-12-01T03:07:20.247+00:00"},"s":"I",
"c":"STORAGE",  "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792040:246821][27:0x7fb184993ac0], txn-
recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global recovery
timestamp: (0, 0)"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:20.247+00:00"},"s":"I",
"c":"STORAGE",  "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792040:247571][27:0x7fb184993ac0], txn-
recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global oldest
timestamp: (0, 0)"}}
myapp            | [nodemon] forking
myapp            | [nodemon] child pid: 33
mongo            | {"t":{"$date":"2020-12-01T03:07:21.012+00:00"},"s":"I",
"c":"STORAGE",  "id":4795906, "ctx":"initandlisten","msg":"WiredTiger
opened","attr":{"durationMillis":3312}}
mongo            | {"t":{"$date":"2020-12-01T03:07:21.014+00:00"},"s":"I",
"c":"RECOVERY", "id":23987,   "ctx":"initandlisten","msg":"WiredTiger
recoveryTimestamp","attr":{"recoveryTimestamp":{"$timestamp":{"t":0,"i":0}}}}
myapp            | [nodemon] watching 19 files
mongo            | {"t":{"$date":"2020-12-01T03:07:23.499+00:00"},"s":"I",
"c":"STORAGE",  "id":4366408, "ctx":"initandlisten","msg":"No table logging
settings modifications are required for existing WiredTiger
tables","attr":{"loggingEnabled":true}}
mongo            | {"t":{"$date":"2020-12-01T03:07:23.565+00:00"},"s":"I",
"c":"STORAGE",  "id":22262,   "ctx":"initandlisten","msg":"Timestamp monitor
starting"}
mongo            | {"t":{"$date":"2020-12-01T03:07:24.102+00:00"},"s":"W",
"c":"CONTROL",  "id":22120,   "ctx":"initandlisten","msg":"Access control is
not enabled for the database. Read and write access to data and configuration
is unrestricted","tags":["startupWarnings"]}
mongo            | {"t":{"$date":"2020-12-01T03:07:24.102+00:00"},"s":"W",
"c":"CONTROL",  "id":22178,
"ctx":"initandlisten","msg":"/sys/kernel/mm/transparent_hugepage/enabled is
'always'. We suggest setting it to 'never'","tags":["startupWarnings"]}
mongo            | {"t":{"$date":"2020-12-01T03:07:24.333+00:00"},"s":"I",
```

```
"c":"STORAGE",  "id":20320,
"ctx":"initandlisten","msg":"createCollection","attr":{"namespace":"admin.sys
tem.version","uuidDisposition":"provided","uuid":{"uuid":{"$uuid":"f86f0839-
3d5d-483a-a06d-a28f7b0a25b5"}},"options":{"uuid":{"$uuid":"f86f0839-3d5d-
483a-a06d-a28f7b0a25b5"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:25.510+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"initandlisten","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"admin.system.version","index"
:"_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:25.529+00:00"},"s":"I",
"c":"COMMAND",  "id":20459,   "ctx":"initandlisten","msg":"Setting
featureCompatibilityVersion","attr":{"newVersion":"4.4"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:25.736+00:00"},"s":"I",
"c":"STORAGE",  "id":20536,   "ctx":"initandlisten","msg":"Flow Control is
enabled on this deployment"}
mongo            | {"t":{"$date":"2020-12-01T03:07:25.836+00:00"},"s":"I",
"c":"STORAGE",  "id":20320,
"ctx":"initandlisten","msg":"createCollection","attr":{"namespace":"local.sta
rtup_log","uuidDisposition":"generated","uuid":{"uuid":{"$uuid":"ddf589c2-
b80d-4ace-a160-4e6eb1feb9a4"}},"options":{"capped":true,"size":10485760}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:26.627+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"initandlisten","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local.startup_log","index":"_
id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
myapp            | Tue, 01 Dec 2020 03:07:27 GMT myapp:server Listening on
port 3000
mongo            | {"t":{"$date":"2020-12-01T03:07:27.823+00:00"},"s":"I",
"c":"FTDC",     "id":20625,   "ctx":"initandlisten","msg":"Initializing full-
time diagnostic data
capture","attr":{"dataDirectory":"/data/db/diagnostic.data"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:27.932+00:00"},"s":"I",
"c":"NETWORK",  "id":23015,   "ctx":"listener","msg":"Listening
on","attr":{"address":"/tmp/mongodb-27017.sock"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:27.932+00:00"},"s":"I",
"c":"NETWORK",  "id":23015,   "ctx":"listener","msg":"Listening
on","attr":{"address":"127.0.0.1"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:27.932+00:00"},"s":"I",
"c":"NETWORK",  "id":23016,   "ctx":"listener","msg":"Waiting for
connections","attr":{"port":27017,"ssl":"off"}}
mongo            | child process started successfully, parent exiting
mongo            | {"t":{"$date":"2020-12-01T03:07:28.236+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"LogicalSessionCacheRefresh","msg":"Slow
query","attr":{"type":"command","ns":"config.system.sessions","command":{"lis
tIndexes":"system.sessions","cursor":{},"$db":"config"},"numYields":0,"ok":0,
"errMsg":"ns does not exist:
config.system.sessions","errName":"NamespaceNotFound","errCode":26,"reslen":1
34,"locks":{"ReplicationStateTransition":{"acquireCount":{"w":1}},"Global":{"
acquireCount":{"r":1}},"Database":{"acquireCount":{"r":1}},"Collection":{"acq
uireCount":{"r":1}},"Mutex":{"acquireCount":{"r":1}}},"protocol":"op_msg","du
rationMillis":241}}
```

```
mongo            | {"t":{"$date":"2020-12-01T03:07:28.236+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"LogicalSessionCacheReap","msg":"Slow
query","attr":{"type":"command","ns":"config.system.sessions","command":{"lis
tIndexes":"system.sessions","cursor":{},"$db":"config"},"numYields":0,"ok":0,
"errMsg":"ns does not exist:
config.system.sessions","errName":"NamespaceNotFound","errCode":26,"reslen":1
34,"locks":{"ReplicationStateTransition":{"acquireCount":{"w":1}},"Global":{"
acquireCount":{"r":1}},"Database":{"acquireCount":{"r":1}},"Collection":{"acq
uireCount":{"r":1}},"Mutex":{"acquireCount":{"r":1}}},"protocol":"op_msg","du
rationMillis":241}}
mongo            | {"t":{"$date":"2020-12-01T03:07:28.238+00:00"},"s":"I",
"c":"CONTROL",  "id":20712,   "ctx":"LogicalSessionCacheReap","msg":"Sessions
collection is not set up; waiting until next sessions reap
interval","attr":{"error":"NamespaceNotFound: config.system.sessions does not
exist"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:28.248+00:00"},"s":"I",
"c":"STORAGE",  "id":20320,
"ctx":"LogicalSessionCacheRefresh","msg":"createCollection","attr":{"namespac
e":"config.system.sessions","uuidDisposition":"generated","uuid":{"uuid":{"$u
uid":"88b72fa8-8f41-4db9-ba5e-4e36c399bce1"}},"options":{}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:29.643+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"LogicalSessionCacheRefresh","msg":"Index
build: done
building","attr":{"buildUUID":null,"namespace":"config.system.sessions","inde
x":"_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:29.643+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"LogicalSessionCacheRefresh","msg":"Index
build: done
building","attr":{"buildUUID":null,"namespace":"config.system.sessions","inde
x":"lsidTTLIndex","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:29.643+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"LogicalSessionCacheRefresh","msg":"Slow
query","attr":{"type":"command","ns":"config.system.sessions","command":{"cre
ateIndexes":"system.sessions","indexes":[{"key":{"lastUse":1},"name":"lsidTTL
Index","expireAfterSeconds":1800}],"writeConcern":{},"$db":"config"},"numYiel
ds":0,"reslen":114,"locks":{"ParallelBatchWriterMode":{"acquireCount":{"r":5}
},"ReplicationStateTransition":{"acquireCount":{"w":5}},"Global":{"acquireCou
nt":{"r":2,"w":3}},"Database":{"acquireCount":{"r":2,"w":3}},"Collection":{"a
cquireCount":{"r":3,"w":2}},"Mutex":{"acquireCount":{"r":6}}},"flowControl":{
"acquireCount":1,"timeAcquiringMicros":2},"storage":{},"protocol":"op_msg","d
urationMillis":1405}}
mongo            | {"t":{"$date":"2020-12-01T03:07:30.792+00:00"},"s":"I",
"c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"127.0.0.1:43028","connectionId":1,"connectionCoun
t":1}}
mongo            | {"t":{"$date":"2020-12-01T03:07:30.874+00:00"},"s":"I",
"c":"NETWORK",  "id":51800,   "ctx":"conn1","msg":"client
metadata","attr":{"remote":"127.0.0.1:43028","client":"conn1","doc":{"applica
tion":{"name":"MongoDB Shell"},"driver":{"name":"MongoDB Internal
Client","version":"4.4.2"},"os":{"type":"Linux","name":"Ubuntu","architecture
```

```
":"x86_64","version":"18.04"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:30.936+00:00"},"s":"I",
"c":"NETWORK",  "id":22944,   "ctx":"conn1","msg":"Connection
ended","attr":{"remote":"127.0.0.1:43028","connectionId":1,"connectionCount":
0}}
mongo            | {"t":{"$date":"2020-12-01T03:07:31.330+00:00"},"s":"I",
"c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"127.0.0.1:43032","connectionId":2,"connectionCoun
t":1}}
mongo            | {"t":{"$date":"2020-12-01T03:07:31.330+00:00"},"s":"I",
"c":"NETWORK",  "id":51800,   "ctx":"conn2","msg":"client
metadata","attr":{"remote":"127.0.0.1:43032","client":"conn2","doc":{"applica
tion":{"name":"MongoDB Shell"},"driver":{"name":"MongoDB Internal
Client","version":"4.4.2"},"os":{"type":"Linux","name":"Ubuntu","architecture
":"x86_64","version":"18.04"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:31.705+00:00"},"s":"I",
"c":"STORAGE",  "id":20320,
"ctx":"conn2","msg":"createCollection","attr":{"namespace":"admin.system.user
s","uuidDisposition":"generated","uuid":{"uuid":{"$uuid":"c09b90ed-efcd-409b-
8fe3-9af044e85a97"}},"options":{}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:33.433+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn2","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"admin.system.users","index":"
_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:33.433+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn2","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"admin.system.users","index":"
user_1_db_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:33.434+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn2","msg":"Slow
query","attr":{"type":"command","ns":"admin.system.users","appName":"MongoDB
Shell","command":{"insert":"system.users","bypassDocumentValidation":false,"o
rdered":true,"$db":"admin"},"ninserted":1,"keysInserted":2,"numYields":0,"res
len":45,"locks":{"ParallelBatchWriterMode":{"acquireCount":{"r":5}},"Replicat
ionStateTransition":{"acquireCount":{"w":5}},"Global":{"acquireCount":{"r":2,
"w":3}},"Database":{"acquireCount":{"r":2,"W":3}},"Collection":{"acquireCount
":{"r":1,"w":3}},"Mutex":{"acquireCount":{"r":5}}},"flowControl":{"acquireCou
nt":4,"timeAcquiringMicros":6},"storage":{},"protocol":"op_msg","durationMill
is":1729}}
mongo            | {"t":{"$date":"2020-12-01T03:07:33.435+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn2","msg":"Slow
query","attr":{"type":"command","ns":"admin.$cmd","appName":"MongoDB
Shell","command":{"createUser":"root","pwd":"xxx","roles":[{"role":"root","db
":"admin"}],"digestPassword":true,"writeConcern":{"w":"majority","wtimeout":6
00000.0},"lsid":{"id":{"$uuid":"f08459bb-a9bb-4324-9bac-
7940f220c5cd"}},"$db":"admin"},"numYields":0,"reslen":38,"locks":{"ParallelBa
tchWriterMode":{"acquireCount":{"r":6}},"ReplicationStateTransition":{"acquir
eCount":{"w":7}},"Global":{"acquireCount":{"r":3,"w":4}},"Database":{"acquire
Count":{"r":2,"W":4}},"Collection":{"acquireCount":{"r":1,"w":4}},"Mutex":{"a
cquireCount":{"r":6}}},"flowControl":{"acquireCount":4,"timeAcquiringMicros":
```

6},"writeConcern":{"w":"majority","wtimeout":600000,"provenance":"clientSuppl
ied"},"storage":{},"protocol":"op_msg","durationMillis":2024}}mongo
| Successfully added user: {
mongo | "user" : "root",
mongo | "roles" : [
mongo | {
mongo | "role" : "root",
mongo | "db" : "admin"
mongo | }
mongo |]
mongo | }
mongo | Error saving history file: FileOpenFailed Unable to open()
file /home/mongodb/.dbshell: No such file or directory
mongo | {"t":{"$date":"2020-12-01T03:07:34.103+00:00"},"s":"I",
"c":"NETWORK", "id":22944,   "ctx":"conn2","msg":"Connection
ended","attr":{"remote":"127.0.0.1:43032","connectionId":2,"connectionCount":
0}}
mongo |
mongo | /usr/local/bin/docker-entrypoint.sh: running /docker-
entrypoint-initdb.d/mongo-init.js
mongo | {"t":{"$date":"2020-12-01T03:07:34.202+00:00"},"s":"I",
"c":"NETWORK", "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"127.0.0.1:43046","connectionId":3,"connectionCoun
t":1}}
mongo | {"t":{"$date":"2020-12-01T03:07:34.203+00:00"},"s":"I",
"c":"NETWORK", "id":51800,   "ctx":"conn3","msg":"client
metadata","attr":{"remote":"127.0.0.1:43046","client":"conn3","doc":{"applica
tion":{"name":"MongoDB Shell"},"driver":{"name":"MongoDB Internal
Client","version":"4.4.2"},"os":{"type":"Linux","name":"Ubuntu","architecture
":"x86_64","version":"18.04"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:34.361+00:00"},"s":"I",
"c":"COMMAND", "id":518070, "ctx":"conn3","msg":"CMD:
drop","attr":{"namespace":"local_library.book"}}
mongo | {"t":{"$date":"2020-12-01T03:07:34.379+00:00"},"s":"I",
"c":"COMMAND", "id":518070, "ctx":"conn3","msg":"CMD:
drop","attr":{"namespace":"local_library.author"}}
mongo | {"t":{"$date":"2020-12-01T03:07:34.380+00:00"},"s":"I",
"c":"COMMAND", "id":518070, "ctx":"conn3","msg":"CMD:
drop","attr":{"namespace":"local_library.bookinstance"}}
mongo | {"t":{"$date":"2020-12-01T03:07:34.381+00:00"},"s":"I",
"c":"COMMAND", "id":518070, "ctx":"conn3","msg":"CMD:
drop","attr":{"namespace":"local_library.genre"}}
mongo | {"t":{"$date":"2020-12-01T03:07:34.383+00:00"},"s":"I",
"c":"STORAGE", "id":20320,
"ctx":"conn3","msg":"createCollection","attr":{"namespace":"local_library.boo
k","uuidDisposition":"generated","uuid":{"uuid":{"$uuid":"79fb1fc6-9ec4-4a37-
ad89-aaa84d553040"}},"options":{}}}
mongo | {"t":{"$date":"2020-12-01T03:07:35.950+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.book","index":"

```
_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:35.950+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.book","index":"
title_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:35.950+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.book","appName":"MongoDB
Shell","command":{"createIndexes":"book","indexes":[{"key":{"title":1.0},"nam
e":"title_1","unique":true}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":5}},"ReplicationStateTransition"
:{"acquireCount":{"w":5}},"Global":{"acquireCount":{"r":2,"w":3}},"Database":
{"acquireCount":{"r":2,"w":3}},"Collection":{"acquireCount":{"r":3,"w":2}},"M
utex":{"acquireCount":{"r":6}}},"flowControl":{"acquireCount":1,"timeAcquirin
gMicros":1},"storage":{},"protocol":"op_msg","durationMillis":1567}}
mongo            | {"t":{"$date":"2020-12-01T03:07:35.951+00:00"},"s":"I",
"c":"INDEX",    "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"000d9be0-2e10-4403-bc06-
40470c3a801f"}},"namespace":"local_library.book","collectionUUID":{"uuid":{"$
uuid":"79fb1fc6-9ec4-4a37-ad89-
aaa84d553040"}},"indexes":1,"firstIndex":{"name":"summary_1"}}}
mongo-express_1  | Welcome to mongo-express
mongo-express_1  | ------------------------
mongo-express_1  |
mongo-express_1  |
mongo-express_1  | Mongo Express server listening at http://0.0.0.0:8081
mongo-express_1  | Server is open to allow connections from anyone (0.0.0.0)
mongo-express_1  | basicAuth credentials are "admin:pass", it is recommended
you change this in your config.js!
mongo            | {"t":{"$date":"2020-12-01T03:07:36.312+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.book","index":"
summary_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:36.312+00:00"},"s":"I",
"c":"INDEX",    "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"000d9be0-2e10-
4403-bc06-
40470c3a801f"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:36.313+00:00"},"s":"I",
"c":"INDEX",    "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"000d9be0-2e10-4403-bc06-
40470c3a801f"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:36.313+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.book","appName":"MongoDB
Shell","command":{"createIndexes":"book","indexes":[{"key":{"summary":1.0},"n
ame":"summary_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
```

:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acquireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicros":3},"storage":{},"protocol":"op_msg","durationMillis":362}}
mongo | {"t":{"$date":"2020-12-01T03:07:36.314+00:00"},"s":"I",
"c":"INDEX",   "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"6a286f49-157b-44e7-abc3-
5003907b2401"}},"namespace":"local_library.book","collectionUUID":{"uuid":{"$
uuid":"79fb1fc6-9ec4-4a37-ad89-
aaa84d553040"}},"indexes":1,"firstIndex":{"name":"author_1"}}}
mongo-express_1 |
mongo-express_1 | /node_modules/mongodb/lib/server.js:265
mongo-express_1 | process.nextTick(function() { throw err; })
mongo-express_1 | ^
mongo-express_1 | Error [MongoError]: failed to connect to server
[mongo:27017] on first connect
mongo-express_1 | at Pool.<anonymous> (/node_modules/mongodb-
core/lib/topologies/server.js:326:35)
mongo-express_1 | at Pool.emit (events.js:314:20)
mongo-express_1 | at Connection.<anonymous> (/node_modules/mongodb-
core/lib/connection/pool.js:270:12)
mongo-express_1 | at Object.onceWrapper (events.js:421:26)
mongo-express_1 | at Connection.emit (events.js:314:20)
mongo-express_1 | at Socket.<anonymous> (/node_modules/mongodb-
core/lib/connection/connection.js:175:49)
mongo-express_1 | at Object.onceWrapper (events.js:421:26)
mongo-express_1 | at Socket.emit (events.js:314:20)
mongo-express_1 | at emitErrorNT (internal/streams/destroy.js:92:8)
mongo-express_1 | at emitErrorAndCloseNT
(internal/streams/destroy.js:60:3)
mongo | {"t":{"$date":"2020-12-01T03:07:36.828+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.book","index":"
author_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:36.828+00:00"},"s":"I",
"c":"INDEX",   "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"6a286f49-157b-
44e7-abc3-
5003907b2401"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:36.828+00:00"},"s":"I",
"c":"INDEX",   "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"6a286f49-157b-44e7-abc3-
5003907b2401"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:36.829+00:00"},"s":"I",
"c":"COMMAND", "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.book","appName":"MongoDB
Shell","command":{"createIndexes":"book","indexes":[{"key":{"author":1.0},"na
me":"author_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"

:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acquireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicros":4},"storage":{},"protocol":"op_msg","durationMillis":514}}
mongo | {"t":{"$date":"2020-12-01T03:07:36.830+00:00"},"s":"I",
"c":"INDEX",   "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"01731c05-d1dc-4069-84cf-
3203dd52b569"}},"namespace":"local_library.book","collectionUUID":{"uuid":{"$
uuid":"79fb1fc6-9ec4-4a37-ad89-
aaa84d553040"}},"indexes":1,"firstIndex":{"name":"isbn_1"}}}
mongo | {"t":{"$date":"2020-12-01T03:07:37.233+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.book","index":"
isbn_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:37.233+00:00"},"s":"I",
"c":"INDEX",   "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"01731c05-d1dc-
4069-84cf-
3203dd52b569"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:37.233+00:00"},"s":"I",
"c":"INDEX",   "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"01731c05-d1dc-4069-84cf-
3203dd52b569"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:37.233+00:00"},"s":"I",
"c":"COMMAND", "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.book","appName":"MongoDB
Shell","command":{"createIndexes":"book","indexes":[{"key":{"isbn":1.0},"name
":"isbn_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":3},"storage":{},"protocol":"op_msg","durationMillis":403}}
mongo | {"t":{"$date":"2020-12-01T03:07:37.234+00:00"},"s":"I",
"c":"INDEX",   "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"8f4f37b8-1444-4741-8b1b-
502cc905ba55"}},"namespace":"local_library.book","collectionUUID":{"uuid":{"$
uuid":"79fb1fc6-9ec4-4a37-ad89-
aaa84d553040"}},"indexes":1,"firstIndex":{"name":"genre_1"}}}
mongo | {"t":{"$date":"2020-12-01T03:07:38.227+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.book","index":"
genre_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:38.227+00:00"},"s":"I",
"c":"INDEX",   "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"8f4f37b8-1444-
4741-8b1b-
502cc905ba55"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:38.227+00:00"},"s":"I",

```
"c":"INDEX",    "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"8f4f37b8-1444-4741-8b1b-
502cc905ba55"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:38.227+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.book","appName":"MongoDB
Shell","command":{"createIndexes":"book","indexes":[{"key":{"genre":1.0},"nam
e":"genre_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":2},"storage":{},"protocol":"op_msg","durationMillis":993}}
mongo            | {"t":{"$date":"2020-12-01T03:07:38.228+00:00"},"s":"I",
"c":"STORAGE",  "id":20320,
"ctx":"conn3","msg":"createCollection","attr":{"namespace":"local_library.aut
hor","uuidDisposition":"generated","uuid":{"uuid":{"$uuid":"534092bc-7172-
4831-9b91-47672ab33dba"}},"options":{}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.333+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.author","index"
:"_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.334+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.author","index"
:"first_name_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.334+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.author","appName":"MongoD
B
Shell","command":{"createIndexes":"author","indexes":[{"key":{"first_name":1.
0},"name":"first_name_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":5}},"ReplicationStateTransition"
:{"acquireCount":{"w":5}},"Global":{"acquireCount":{"r":2,"w":3}},"Database":
{"acquireCount":{"r":2,"w":3}},"Collection":{"acquireCount":{"r":4,"w":2}},"M
utex":{"acquireCount":{"r":5}}},"flowControl":{"acquireCount":1,"timeAcquirin
gMicros":1},"storage":{},"protocol":"op_msg","durationMillis":1105}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.334+00:00"},"s":"I",
"c":"INDEX",    "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"8611f8b1-3388-4bca-8cd6-
28eb930a43da"}},"namespace":"local_library.author","collectionUUID":{"uuid":{
"$uuid":"534092bc-7172-4831-9b91-
47672ab33dba"}},"indexes":1,"firstIndex":{"name":"family_name_1"}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.794+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.author","index"
:"family_name_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.794+00:00"},"s":"I",
```

```
"c":"INDEX",    "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"8611f8b1-3388-
4bca-8cd6-
28eb930a43da"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.794+00:00"},"s":"I",
"c":"INDEX",    "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"8611f8b1-3388-4bca-8cd6-
28eb930a43da"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.794+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.author","appName":"MongoD
B
Shell","command":{"createIndexes":"author","indexes":[{"key":{"family_name":1
.0},"name":"family_name_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":3},"storage":{},"protocol":"op_msg","durationMillis":460}}
mongo            | {"t":{"$date":"2020-12-01T03:07:39.795+00:00"},"s":"I",
"c":"INDEX",    "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"bbea2410-f9ee-4155-84b6-
561cce199e5f"}},"namespace":"local_library.author","collectionUUID":{"uuid":{
"$uuid":"534092bc-7172-4831-9b91-
47672ab33dba"}},"indexes":1,"firstIndex":{"name":"d_birth_1"}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.232+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.author","index"
:"d_birth_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.232+00:00"},"s":"I",
"c":"INDEX",    "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"bbea2410-f9ee-
4155-84b6-
561cce199e5f"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.232+00:00"},"s":"I",
"c":"INDEX",    "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"bbea2410-f9ee-4155-84b6-
561cce199e5f"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.232+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.author","appName":"MongoD
B
Shell","command":{"createIndexes":"author","indexes":[{"key":{"d_birth":1.0},
"name":"d_birth_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
```

```
s":3},"storage":{},"protocol":"op_msg","durationMillis":436}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.233+00:00"},"s":"I",
"c":"INDEX",    "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"56a5450e-8e45-49c3-b366-
f33a9dad353d"}},"namespace":"local_library.author","collectionUUID":{"uuid":{
"$uuid":"534092bc-7172-4831-9b91-
47672ab33dba"}},"indexes":1,"firstIndex":{"name":"d_death_1"}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.865+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.author","index"
:"d_death_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.865+00:00"},"s":"I",
"c":"INDEX",    "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"56a5450e-8e45-
49c3-b366-
f33a9dad353d"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.865+00:00"},"s":"I",
"c":"INDEX",    "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"56a5450e-8e45-49c3-b366-
f33a9dad353d"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.866+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.author","appName":"MongoD
B
Shell","command":{"createIndexes":"author","indexes":[{"key":{"d_death":1.0},
"name":"d_death_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":3},"storage":{},"protocol":"op_msg","durationMillis":632}}
mongo            | {"t":{"$date":"2020-12-01T03:07:40.866+00:00"},"s":"I",
"c":"STORAGE",  "id":20320,
"ctx":"conn3","msg":"createCollection","attr":{"namespace":"local_library.boo
kinstance","uuidDisposition":"generated","uuid":{"uuid":{"$uuid":"fd605567-
f031-4c6b-b2c9-5719724e20d3"}},"options":{}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:42.479+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.bookinstance","
index":"_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:42.479+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.bookinstance","
index":"book_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:42.480+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.bookinstance","appName":"
MongoDB
Shell","command":{"createIndexes":"bookinstance","indexes":[{"key":{"book":1.
```

0},"name":"book_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":5}},"ReplicationStateTransition"
:{"acquireCount":{"w":5}},"Global":{"acquireCount":{"r":2,"w":3}},"Database":
{"acquireCount":{"r":2,"w":3}},"Collection":{"acquireCount":{"r":4,"w":2}},"M
utex":{"acquireCount":{"r":5}}},"flowControl":{"acquireCount":1,"timeAcquirin
gMicros":2},"storage":{},"protocol":"op_msg","durationMillis":1613}}
mongo | {"t":{"$date":"2020-12-01T03:07:42.480+00:00"},"s":"I",
"c":"INDEX",   "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"9d2f79bb-48e2-465a-ad7d-
4c9bab953bce"}},"namespace":"local_library.bookinstance","collectionUUID":{"u
uid":{"$uuid":"fd605567-f031-4c6b-b2c9-
5719724e20d3"}},"indexes":1,"firstIndex":{"name":"imprint_1"}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.499+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.bookinstance","
index":"imprint_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.499+00:00"},"s":"I",
"c":"INDEX",   "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"9d2f79bb-48e2-
465a-ad7d-
4c9bab953bce"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.499+00:00"},"s":"I",
"c":"INDEX",   "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"9d2f79bb-48e2-465a-ad7d-
4c9bab953bce"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.499+00:00"},"s":"I",
"c":"COMMAND", "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.bookinstance","appName":"
MongoDB
Shell","command":{"createIndexes":"bookinstance","indexes":[{"key":{"imprint"
:1.0},"name":"imprint_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":4},"storage":{},"protocol":"op_msg","durationMillis":1018}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.500+00:00"},"s":"I",
"c":"INDEX",   "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"3748cff2-eb96-4e3c-b346-
781b8d5ed28f"}},"namespace":"local_library.bookinstance","collectionUUID":{"u
uid":{"$uuid":"fd605567-f031-4c6b-b2c9-
5719724e20d3"}},"indexes":1,"firstIndex":{"name":"due_back_1"}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.838+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.bookinstance","
index":"due_back_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.838+00:00"},"s":"I",
"c":"INDEX",   "id":20440,   "ctx":"conn3","msg":"Index build: waiting for

index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"3748cff2-eb96-4e3c-b346-
781b8d5ed28f"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.838+00:00"},"s":"I",
"c":"INDEX",   "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"3748cff2-eb96-4e3c-b346-
781b8d5ed28f"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.838+00:00"},"s":"I",
"c":"COMMAND", "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.bookinstance","appName":"
MongoDB
Shell","command":{"createIndexes":"bookinstance","indexes":[{"key":{"due_back
":1.0},"name":"due_back_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":1},"storage":{},"protocol":"op_msg","durationMillis":338}}
mongo | {"t":{"$date":"2020-12-01T03:07:43.839+00:00"},"s":"I",
"c":"INDEX",   "id":20438,   "ctx":"conn3","msg":"Index build:
registering","attr":{"buildUUID":{"uuid":{"$uuid":"0b4f5bd2-80d9-4da4-b328-
e1776265d13c"}},"namespace":"local_library.bookinstance","collectionUUID":{"u
uid":{"$uuid":"fd605567-f031-4c6b-b2c9-
5719724e20d3"}},"indexes":1,"firstIndex":{"name":"status_1"}}}
mongo | {"t":{"$date":"2020-12-01T03:07:44.201+00:00"},"s":"I",
"c":"INDEX",   "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.bookinstance","
index":"status_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:44.201+00:00"},"s":"I",
"c":"INDEX",   "id":20440,   "ctx":"conn3","msg":"Index build: waiting for
index build to complete","attr":{"buildUUID":{"uuid":{"$uuid":"0b4f5bd2-80d9-
4da4-b328-
e1776265d13c"}},"deadline":{"$date":{"$numberLong":"9223372036854775807"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:44.201+00:00"},"s":"I",
"c":"INDEX",   "id":20447,   "ctx":"conn3","msg":"Index build:
completed","attr":{"buildUUID":{"uuid":{"$uuid":"0b4f5bd2-80d9-4da4-b328-
e1776265d13c"}}}}
mongo | {"t":{"$date":"2020-12-01T03:07:44.202+00:00"},"s":"I",
"c":"COMMAND", "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.bookinstance","appName":"
MongoDB
Shell","command":{"createIndexes":"bookinstance","indexes":[{"key":{"status":
1.0},"name":"status_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":3}},"ReplicationStateTransition"
:{"acquireCount":{"w":4}},"Global":{"acquireCount":{"w":4}},"Database":{"acqu
ireCount":{"w":3}},"Collection":{"acquireCount":{"r":1,"w":1,"W":1}},"Mutex":
{"acquireCount":{"r":3}}},"flowControl":{"acquireCount":3,"timeAcquiringMicro
s":2},"storage":{},"protocol":"op_msg","durationMillis":362}}

```
mongo            | {"t":{"$date":"2020-12-01T03:07:44.203+00:00"},"s":"I",
"c":"STORAGE",  "id":20320,
"ctx":"conn3","msg":"createCollection","attr":{"namespace":"local_library.gen
re","uuidDisposition":"generated","uuid":{"uuid":{"$uuid":"e2e3bdc3-8ef3-
4bcc-a2d0-bf0ba50df8cd"}},"options":{}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.427+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.genre","index":
"_id_","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.427+00:00"},"s":"I",
"c":"INDEX",    "id":20345,   "ctx":"conn3","msg":"Index build: done
building","attr":{"buildUUID":null,"namespace":"local_library.genre","index":
"name_1","commitTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.427+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.genre","appName":"MongoDB
Shell","command":{"createIndexes":"genre","indexes":[{"key":{"name":1.0},"nam
e":"name_1"}],"lsid":{"id":{"$uuid":"f40616aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"numYields":0,"reslen":114,"locks":{"P
arallelBatchWriterMode":{"acquireCount":{"r":5}},"ReplicationStateTransition"
:{"acquireCount":{"w":5}},"Global":{"acquireCount":{"r":2,"w":3}},"Database":
{"acquireCount":{"r":2,"w":3}},"Collection":{"acquireCount":{"r":4,"w":2}},"M
utex":{"acquireCount":{"r":5}}},"flowControl":{"acquireCount":1,"timeAcquirin
gMicros":1},"storage":{},"protocol":"op_msg","durationMillis":1224}}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.617+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"conn3","msg":"Slow
query","attr":{"type":"command","ns":"local_library.book","appName":"MongoDB
Shell","command":{"insert":"book","ordered":true,"lsid":{"id":{"$uuid":"f4061
6aa-47a2-400a-80d6-
d8e3e8f97a2a"}},"$db":"local_library"},"ninserted":1,"keysInserted":6,"numYie
lds":0,"reslen":45,"locks":{"ParallelBatchWriterMode":{"acquireCount":{"r":1}
},"ReplicationStateTransition":{"acquireCount":{"w":1}},"Global":{"acquireCou
nt":{"w":1}},"Database":{"acquireCount":{"w":1}},"Collection":{"acquireCount"
:{"w":1}},"Mutex":{"acquireCount":{"r":1}}},"flowControl":{"acquireCount":1,"
timeAcquiringMicros":1},"storage":{},"protocol":"op_msg","durationMillis":124
}}
mongo            | [
mongo            |      false,
mongo            |      false,
mongo            |      false,
mongo            |      false,
mongo            |      {
mongo            |              "createdCollectionAutomatically" : true,
mongo            |              "numIndexesBefore" : 1,
mongo            |              "numIndexesAfter" : 2,
mongo            |              "ok" : 1
mongo            |      },
mongo            |      {
mongo            |              "createdCollectionAutomatically" : false,
mongo            |              "numIndexesBefore" : 2,
```

```
mongo           |              "numIndexesAfter" : 3,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : false,
mongo           |              "numIndexesBefore" : 3,
mongo           |              "numIndexesAfter" : 4,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : false,
mongo           |              "numIndexesBefore" : 4,
mongo           |              "numIndexesAfter" : 5,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : false,
mongo           |              "numIndexesBefore" : 5,
mongo           |              "numIndexesAfter" : 6,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : true,
mongo           |              "numIndexesBefore" : 1,
mongo           |              "numIndexesAfter" : 2,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : false,
mongo           |              "numIndexesBefore" : 2,
mongo           |              "numIndexesAfter" : 3,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : false,
mongo           |              "numIndexesBefore" : 3,
mongo           |              "numIndexesAfter" : 4,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : false,
mongo           |              "numIndexesBefore" : 4,
mongo           |              "numIndexesAfter" : 5,
mongo           |              "ok" : 1
mongo           |      },
mongo           |      {
mongo           |              "createdCollectionAutomatically" : true,
mongo           |              "numIndexesBefore" : 1,
mongo           |              "numIndexesAfter" : 2,
mongo           |              "ok" : 1
```

```
mongo           |       },
mongo           |       {
mongo           |               "createdCollectionAutomatically" : false,
mongo           |               "numIndexesBefore" : 2,
mongo           |               "numIndexesAfter" : 3,
mongo           |               "ok" : 1
mongo           |       },
mongo           |       {
mongo           |               "createdCollectionAutomatically" : false,
mongo           |               "numIndexesBefore" : 3,
mongo           |               "numIndexesAfter" : 4,
mongo           |               "ok" : 1
mongo           |       },
mongo           |       {
mongo           |               "createdCollectionAutomatically" : false,
mongo           |               "numIndexesBefore" : 4,
mongo           |               "numIndexesAfter" : 5,
mongo           |               "ok" : 1
mongo           |       },
mongo           |       {
mongo           |               "createdCollectionAutomatically" : true,
mongo           |               "numIndexesBefore" : 1,
mongo           |               "numIndexesAfter" : 2,
mongo           |               "ok" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
mongo           |               "nInserted" : 1
mongo           |       },
mongo           |       {
```

```
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
mongo               |           {
mongo               |                   "nInserted" : 1
mongo               |           },
```

```
mongo            |      {
mongo            |              "nInserted" : 1
mongo            |      }
mongo            | ]
mongo            | {"t":{"$date":"2020-12-01T03:07:45.641+00:00"},"s":"I",
"c":"NETWORK",  "id":22944,   "ctx":"conn3","msg":"Connection
ended","attr":{"remote":"127.0.0.1:43046","connectionId":3,"connectionCount":
0}}
mongo            |
mongo            |
mongo            |
mongo            | {"t":{"$date":"2020-12-01T03:07:45.658+00:00"},"s":"I",
"c":"CONTROL",  "id":20698,   "ctx":"main","msg":"***** SERVER RESTARTED
*****"}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.661+00:00"},"s":"I",
"c":"CONTROL",  "id":23285,   "ctx":"main","msg":"Automatically disabling TLS
1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.663+00:00"},"s":"W",
"c":"ASIO",     "id":22601,   "ctx":"main","msg":"No TransportLayer
configured during NetworkInterface startup"}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.664+00:00"},"s":"I",
"c":"NETWORK",  "id":4648601, "ctx":"main","msg":"Implicit TCP FastOpen
unavailable. If TCP FastOpen is required, set tcpFastOpenServer,
tcpFastOpenClient, and tcpFastOpenQueueSize."}
mongo            | killing process with pid: 27
mongo            | {"t":{"$date":"2020-12-01T03:07:45.664+00:00"},"s":"I",
"c":"CONTROL",  "id":23377,   "ctx":"SignalHandler","msg":"Received
signal","attr":{"signal":15,"error":"Terminated"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.664+00:00"},"s":"I",
"c":"CONTROL",  "id":23378,   "ctx":"SignalHandler","msg":"Signal was sent by
kill(2)","attr":{"pid":87,"uid":999}}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.800+00:00"},"s":"I",
"c":"CONTROL",  "id":23381,   "ctx":"SignalHandler","msg":"will terminate
after current cmd ends"}
mongo            | {"t":{"$date":"2020-12-01T03:07:45.824+00:00"},"s":"I",
"c":"REPL",     "id":4784900, "ctx":"SignalHandler","msg":"Stepping down the
ReplicationCoordinator for shutdown","attr":{"waitTimeMillis":10000}}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.063+00:00"},"s":"I",
"c":"COMMAND",  "id":4784901, "ctx":"SignalHandler","msg":"Shutting down the
MirrorMaestro"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.063+00:00"},"s":"I",
"c":"SHARDING", "id":4784902, "ctx":"SignalHandler","msg":"Shutting down the
WaitForMajorityService"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.107+00:00"},"s":"I",
"c":"CONTROL",  "id":4784903, "ctx":"SignalHandler","msg":"Shutting down the
LogicalSessionCache"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.108+00:00"},"s":"I",
"c":"NETWORK",  "id":20562,   "ctx":"SignalHandler","msg":"Shutdown: going to
close listening sockets"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.108+00:00"},"s":"I",
```

```
"c":"NETWORK",  "id":23017,   "ctx":"listener","msg":"removing socket
file","attr":{"path":"/tmp/mongodb-27017.sock"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.108+00:00"},"s":"I",
"c":"NETWORK",  "id":4784905, "ctx":"SignalHandler","msg":"Shutting down the
global connection pool"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.108+00:00"},"s":"I",
"c":"STORAGE",  "id":4784906, "ctx":"SignalHandler","msg":"Shutting down the
FlowControlTicketholder"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.108+00:00"},"s":"I",
"c":"-",        "id":20520,   "ctx":"SignalHandler","msg":"Stopping further
Flow Control ticket acquisitions."}mongo            | {"t":{"$date":"2020-12-
01T03:07:46.114+00:00"},"s":"I",  "c":"STORAGE",  "id":4784908,
"ctx":"SignalHandler","msg":"Shutting down the
PeriodicThreadToAbortExpiredTransactions"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.114+00:00"},"s":"I",
"c":"STORAGE",  "id":4784934, "ctx":"SignalHandler","msg":"Shutting down the
PeriodicThreadToDecreaseSnapshotHistoryCachePressure"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"REPL",     "id":4784909, "ctx":"SignalHandler","msg":"Shutting down the
ReplicationCoordinator"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"SHARDING", "id":4784910, "ctx":"SignalHandler","msg":"Shutting down the
ShardingInitializationMongoD"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"REPL",     "id":4784911, "ctx":"SignalHandler","msg":"Enqueuing the
ReplicationStateTransitionLock for shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"-",        "id":4784912, "ctx":"SignalHandler","msg":"Killing all
operations for shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"-",        "id":4695300, "ctx":"SignalHandler","msg":"Interrupted all
currently running operations","attr":{"opsKilled":3}}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"COMMAND",  "id":4784913, "ctx":"SignalHandler","msg":"Shutting down all
open transactions"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"REPL",     "id":4784914, "ctx":"SignalHandler","msg":"Acquiring the
ReplicationStateTransitionLock for shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"INDEX",    "id":4784915, "ctx":"SignalHandler","msg":"Shutting down the
IndexBuildsCoordinator"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"REPL",     "id":4784916, "ctx":"SignalHandler","msg":"Reacquiring the
ReplicationStateTransitionLock for shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.115+00:00"},"s":"I",
"c":"REPL",     "id":4784917, "ctx":"SignalHandler","msg":"Attempting to mark
clean shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.116+00:00"},"s":"I",
"c":"NETWORK",  "id":4784918, "ctx":"SignalHandler","msg":"Shutting down the
ReplicaSetMonitor"}
```

```
mongo            | {"t":{"$date":"2020-12-01T03:07:46.117+00:00"},"s":"I",
"c":"SHARDING", "id":4784921, "ctx":"SignalHandler","msg":"Shutting down the
MigrationUtilExecutor"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.117+00:00"},"s":"I",
"c":"CONTROL",  "id":4784925, "ctx":"SignalHandler","msg":"Shutting down free
monitoring"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.117+00:00"},"s":"I",
"c":"CONTROL",  "id":20609,   "ctx":"SignalHandler","msg":"Shutting down free
monitoring"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.117+00:00"},"s":"I",
"c":"FTDC",     "id":4784926, "ctx":"SignalHandler","msg":"Shutting down
full-time data capture"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.117+00:00"},"s":"I",
"c":"FTDC",     "id":20626,   "ctx":"SignalHandler","msg":"Shutting down
full-time diagnostic data capture"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.120+00:00"},"s":"I",
"c":"STORAGE",  "id":4784927, "ctx":"SignalHandler","msg":"Shutting down the
HealthLog"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.120+00:00"},"s":"I",
"c":"STORAGE",  "id":4784929, "ctx":"SignalHandler","msg":"Acquiring the
global lock for shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.120+00:00"},"s":"I",
"c":"STORAGE",  "id":4784930, "ctx":"SignalHandler","msg":"Shutting down the
storage engine"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.120+00:00"},"s":"I",
"c":"STORAGE",  "id":20282,   "ctx":"SignalHandler","msg":"Deregistering all
the collections"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.120+00:00"},"s":"I",
"c":"STORAGE",  "id":22261,   "ctx":"SignalHandler","msg":"Timestamp monitor
shutting down"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.121+00:00"},"s":"I",
"c":"STORAGE",  "id":22317,   "ctx":"SignalHandler","msg":"WiredTigerKVEngine
shutting down"}
mongo-express_1  | Tue Dec  1 03:07:46 UTC 2020 retrying to connect to
mongo:27017 (2/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo            | {"t":{"$date":"2020-12-01T03:07:46.221+00:00"},"s":"I",
"c":"STORAGE",  "id":22318,   "ctx":"SignalHandler","msg":"Shutting down
session sweeper thread"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.222+00:00"},"s":"I",
"c":"STORAGE",  "id":22319,   "ctx":"SignalHandler","msg":"Finished shutting
down session sweeper thread"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.222+00:00"},"s":"I",
"c":"STORAGE",  "id":22320,   "ctx":"SignalHandler","msg":"Shutting down
journal flusher thread"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.222+00:00"},"s":"I",
"c":"STORAGE",  "id":22321,   "ctx":"SignalHandler","msg":"Finished shutting
down journal flusher thread"}
```

```
mongo            | {"t":{"$date":"2020-12-01T03:07:46.222+00:00"},"s":"I",
"c":"STORAGE",  "id":22322,   "ctx":"SignalHandler","msg":"Shutting down
checkpoint thread"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.222+00:00"},"s":"I",
"c":"STORAGE",  "id":22323,   "ctx":"SignalHandler","msg":"Finished shutting
down checkpoint thread"}
mongo            | {"t":{"$date":"2020-12-01T03:07:46.222+00:00"},"s":"I",
"c":"STORAGE",  "id":4795902, "ctx":"SignalHandler","msg":"Closing
WiredTiger","attr":{"closeConfig":"leak_memory=true,"}}
mongo-express_1  | Tue Dec  1 03:07:47 UTC 2020 retrying to connect to
mongo:27017 (3/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Tue Dec  1 03:07:48 UTC 2020 retrying to connect to
mongo:27017 (4/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Tue Dec  1 03:07:49 UTC 2020 retrying to connect to
mongo:27017 (5/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo-express_1  | Welcome to mongo-express
mongo-express_1  | ------------------------
mongo-express_1  |
mongo-express_1  |
mongo-express_1  | Mongo Express server listening at http://0.0.0.0:8081
mongo-express_1  | Server is open to allow connections from anyone (0.0.0.0)
mongo-express_1  | basicAuth credentials are "admin:pass", it is recommended
you change this in your config.js!
mongo-express_1  |
mongo-express_1  | /node_modules/mongodb/lib/server.js:265
mongo-express_1  |         process.nextTick(function() { throw err; })
mongo-express_1  |                                         ^
mongo-express_1  | Error [MongoError]: failed to connect to server
[mongo:27017] on first connect
mongo-express_1  |     at Pool.<anonymous> (/node_modules/mongodb-
core/lib/topologies/server.js:326:35)
mongo-express_1  |     at Pool.emit (events.js:314:20)
mongo-express_1  |     at Connection.<anonymous> (/node_modules/mongodb-
core/lib/connection/pool.js:270:12)
mongo-express_1  |     at Object.onceWrapper (events.js:421:26)
mongo-express_1  |     at Connection.emit (events.js:314:20)
mongo-express_1  |     at Socket.<anonymous> (/node_modules/mongodb-
core/lib/connection/connection.js:175:49)
mongo-express_1  |     at Object.onceWrapper (events.js:421:26)
mongo-express_1  |     at Socket.emit (events.js:314:20)
mongo-express_1  |     at emitErrorNT (internal/streams/destroy.js:92:8)
```

```
mongo-express_1  |     at emitErrorAndCloseNT
(internal/streams/destroy.js:60:3)
mongo            | {"t":{"$date":"2020-12-01T03:07:52.101+00:00"},"s":"I",
"c":"STORAGE",  "id":4795901, "ctx":"SignalHandler","msg":"WiredTiger
closed","attr":{"durationMillis":5879}}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.101+00:00"},"s":"I",
"c":"STORAGE",  "id":22279,   "ctx":"SignalHandler","msg":"shutdown: removing
fs lock..."}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.101+00:00"},"s":"I",
"c":"-",        "id":4784931, "ctx":"SignalHandler","msg":"Dropping the scope
cache for shutdown"}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.101+00:00"},"s":"I",
"c":"CONTROL",  "id":20565,   "ctx":"SignalHandler","msg":"Now exiting"}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.101+00:00"},"s":"I",
"c":"CONTROL",  "id":23138,   "ctx":"SignalHandler","msg":"Shutting
down","attr":{"exitCode":0}}
mongo            |
mongo            | MongoDB init process complete; ready for start up.
mongo            |
mongo            | {"t":{"$date":"2020-12-01T03:07:52.687+00:00"},"s":"I",
"c":"CONTROL",  "id":23285,   "ctx":"main","msg":"Automatically disabling TLS
1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.688+00:00"},"s":"W",
"c":"ASIO",     "id":22601,   "ctx":"main","msg":"No TransportLayer
configured during NetworkInterface startup"}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.689+00:00"},"s":"I",
"c":"NETWORK",  "id":4648601, "ctx":"main","msg":"Implicit TCP FastOpen
unavailable. If TCP FastOpen is required, set tcpFastOpenServer,
tcpFastOpenClient, and tcpFastOpenQueueSize."}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.689+00:00"},"s":"I",
"c":"STORAGE",  "id":4615611, "ctx":"initandlisten","msg":"MongoDB
starting","attr":{"pid":1,"port":27017,"dbPath":"/data/db","architecture":"64
-bit","host":"82bed784144a"}}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.689+00:00"},"s":"I",
"c":"CONTROL",  "id":23403,   "ctx":"initandlisten","msg":"Build
Info","attr":{"buildInfo":{"version":"4.4.2","gitVersion":"15e73dc5738d2278b6
88f8929aee605fe4279b0e","openSSLVersion":"OpenSSL 1.1.1  11 Sep
2018","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu180
4","distarch":"x86_64","target_arch":"x86_64"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.689+00:00"},"s":"I",
"c":"CONTROL",  "id":51765,   "ctx":"initandlisten","msg":"Operating
System","attr":{"os":{"name":"Ubuntu","version":"18.04"}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.689+00:00"},"s":"I",
"c":"CONTROL",  "id":21951,   "ctx":"initandlisten","msg":"Options set by
command
line","attr":{"options":{"net":{"bindIp":"*"},"security":{"authorization":"en
abled"}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:52.691+00:00"},"s":"I",
"c":"STORAGE",  "id":22270,   "ctx":"initandlisten","msg":"Storage engine to
use detected by data
```

files","attr":{"dbpath":"/data/db","storageEngine":"wiredTiger"}}
mongo | {"t":{"$date":"2020-12-01T03:07:52.691+00:00"},"s":"I",
"c":"STORAGE", "id":22297,   "ctx":"initandlisten","msg":"Using the XFS
filesystem is strongly recommended with the WiredTiger storage engine. See
http://dochub.mongodb.org/core/prodnotes-
filesystem","tags":["startupWarnings"]}
mongo | {"t":{"$date":"2020-12-01T03:07:52.691+00:00"},"s":"I",
"c":"STORAGE", "id":22315,   "ctx":"initandlisten","msg":"Opening
WiredTiger","attr":{"config":"create,cache_size=1823M,session_max=33000,evict
ion=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(en
abled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_i
dle_time=100000,close_scan_interval=10,close_handle_minimum=250),statistics_l
og=(wait=0),verbose=[recovery_progress,checkpoint_progress,compact_progress],
"}}
express_app3_mongo-express_1 exited with code 1
mongo | {"t":{"$date":"2020-12-01T03:07:53.787+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792073:786997][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 1 through 2"}}
mongo | {"t":{"$date":"2020-12-01T03:07:53.882+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792073:882745][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 2 through 2"}}
mongo | {"t":{"$date":"2020-12-01T03:07:54.106+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792074:106665][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Main recovery loop:
starting at 1/95360 to 2/256"}}
mongo | {"t":{"$date":"2020-12-01T03:07:54.254+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792074:254270][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 1 through 2"}}
mongo | {"t":{"$date":"2020-12-01T03:07:54.504+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792074:504793][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 2 through 2"}}
mongo | {"t":{"$date":"2020-12-01T03:07:54.614+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792074:614782][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global recovery
timestamp: (0, 0)"}}
mongo | {"t":{"$date":"2020-12-01T03:07:54.614+00:00"},"s":"I",
"c":"STORAGE", "id":22430,   "ctx":"initandlisten","msg":"WiredTiger
message","attr":{"message":"[1606792074:614908][1:0x7efc5320bac0], txn-
recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global oldest
timestamp: (0, 0)"}}
myapp | MongoDB connection error: MongooseServerSelectionError:
connect ECONNREFUSED 172.22.0.3:27017
myapp | at NativeConnection.Connection.openUri
(/node_modules/mongoose/lib/connection.js:832:32)

```
myapp            |     at /node_modules/mongoose/lib/index.js:345:10
myapp            |     at
/node_modules/mongoose/lib/helpers/promiseOrCallback.js:31:5
myapp            |     at new Promise (<anonymous>)
myapp            |     at promiseOrCallback
(/node_modules/mongoose/lib/helpers/promiseOrCallback.js:30:10)
myapp            |     at Mongoose._promiseOrCallback
(/node_modules/mongoose/lib/index.js:1135:10)
myapp            |     at Mongoose.connect
(/node_modules/mongoose/lib/index.js:344:20)
myapp            |     at Object.<anonymous> (/usr/src/app/app.js:16:10)
myapp            |     at Module._compile
(internal/modules/cjs/loader.js:999:30)
myapp            |     at Object.Module._extensions..js
(internal/modules/cjs/loader.js:1027:10)
myapp            |     at Module.load (internal/modules/cjs/loader.js:863:32)
myapp            |     at Function.Module._load
(internal/modules/cjs/loader.js:708:14)
myapp            |     at Module.require
(internal/modules/cjs/loader.js:887:19)
myapp            |     at require (internal/modules/cjs/helpers.js:74:18)
myapp            |     at Object.<anonymous> (/usr/src/app/bin/www:7:11)
myapp            |     at Module._compile
(internal/modules/cjs/loader.js:999:30) {
myapp            |   reason: TopologyDescription {
myapp            |     type: 'Unknown',
myapp            |     setName: null,
myapp            |     maxSetVersion: null,
myapp            |     maxElectionId: null,
myapp            |     servers: Map { 'mongo:27017' => [ServerDescription] },
myapp            |     stale: false,
myapp            |     compatible: true,
myapp            |     compatibilityError: null,
myapp            |     logicalSessionTimeoutMinutes: null,
myapp            |     heartbeatFrequencyMS: 10000,
myapp            |     localThresholdMS: 15,
myapp            |     commonWireVersion: null
myapp            |   }
myapp            | }
myapp            | (node:33) UnhandledPromiseRejectionWarning:
MongooseServerSelectionError: connect ECONNREFUSED 172.22.0.3:27017
myapp            |     at NativeConnection.Connection.openUri
(/node_modules/mongoose/lib/connection.js:832:32)
myapp            |     at /node_modules/mongoose/lib/index.js:345:10
myapp            |     at
/node_modules/mongoose/lib/helpers/promiseOrCallback.js:31:5
myapp            |     at new Promise (<anonymous>)
myapp            |     at promiseOrCallback
(/node_modules/mongoose/lib/helpers/promiseOrCallback.js:30:10)
myapp            |     at Mongoose._promiseOrCallback
```

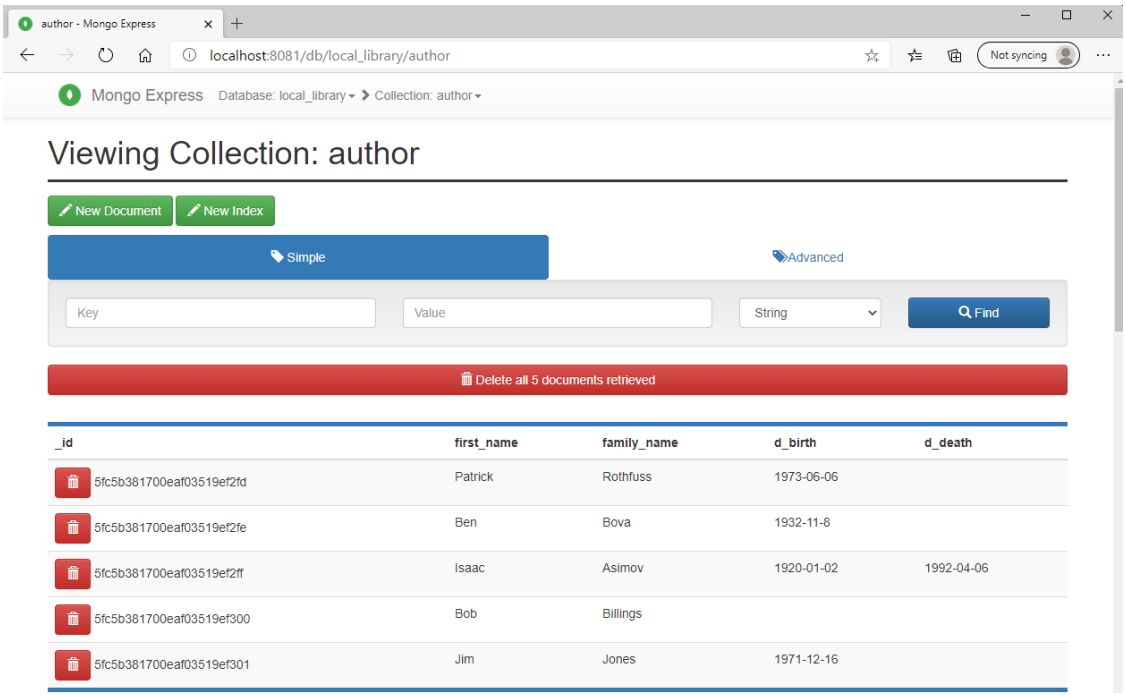
```
(/node_modules/mongoose/lib/index.js:1135:10)
myapp            |     at Mongoose.connect
(/node_modules/mongoose/lib/index.js:344:20)
myapp            |     at Object.<anonymous> (/usr/src/app/app.js:16:10)
myapp            |     at Module._compile
(internal/modules/cjs/loader.js:999:30)
myapp            |     at Object.Module._extensions..js
(internal/modules/cjs/loader.js:1027:10)
myapp            |     at Module.load (internal/modules/cjs/loader.js:863:32)
myapp            |     at Function.Module._load
(internal/modules/cjs/loader.js:708:14)
myapp            |     at Module.require
(internal/modules/cjs/loader.js:887:19)
myapp            |     at require (internal/modules/cjs/helpers.js:74:18)
myapp            |     at Object.<anonymous> (/usr/src/app/bin/www:7:11)
myapp            |     at Module._compile
(internal/modules/cjs/loader.js:999:30)
myapp            | (node:33) UnhandledPromiseRejectionWarning: Unhandled
promise rejection. This error originated either by throwing inside of an
async function without a catch block, or
by rejecting a promise which was not handled with .catch(). To terminate the
node process on unhandled promise rejection, use the CLI flag `--unhandled-
rejections=strict` (see
https://nodejs.org/api/cli.html#cli_unhandled_rejections_mode). (rejection
id: 1)
myapp            | (node:33) [DEP0018] DeprecationWarning: Unhandled promise
rejections are deprecated. In the future, promise rejections that are not
handled will terminate the Node.js process with a non-zero exit code.
mongo-express_1  | Tue Dec  1 03:07:59 UTC 2020 retrying to connect to
mongo:27017 (2/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
Connection refused
mongo            | {"t":{"$date":"2020-12-01T03:07:59.524+00:00"},"s":"I",
"c":"STORAGE",  "id":4795906, "ctx":"initandlisten","msg":"WiredTiger
opened","attr":{"durationMillis":6833}}
mongo            | {"t":{"$date":"2020-12-01T03:07:59.524+00:00"},"s":"I",
"c":"RECOVERY", "id":23987,   "ctx":"initandlisten","msg":"WiredTiger
recoveryTimestamp","attr":{"recoveryTimestamp":{"$timestamp":{"t":0,"i":0}}}}
mongo            | {"t":{"$date":"2020-12-01T03:07:59.525+00:00"},"s":"I",
"c":"STORAGE",  "id":4366408, "ctx":"initandlisten","msg":"No table logging
settings modifications are required for existing WiredTiger
tables","attr":{"loggingEnabled":true}}
mongo            | {"t":{"$date":"2020-12-01T03:07:59.573+00:00"},"s":"I",
"c":"STORAGE",  "id":22262,   "ctx":"initandlisten","msg":"Timestamp monitor
starting"}
mongo-express_1  | Tue Dec  1 03:08:00 UTC 2020 retrying to connect to
mongo:27017 (3/5)
mongo-express_1  | /docker-entrypoint.sh: connect: Connection refused
mongo-express_1  | /docker-entrypoint.sh: line 14: /dev/tcp/mongo/27017:
```

```
Connection refused
mongo            | {"t":{"$date":"2020-12-01T03:08:00.547+00:00"},"s":"W",
"c":"CONTROL",  "id":22178,
"ctx":"initandlisten","msg":"/sys/kernel/mm/transparent_hugepage/enabled is
'always'. We suggest setting it to 'never'","tags":["startupWarnings"]}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.551+00:00"},"s":"I",
"c":"STORAGE",  "id":20536,   "ctx":"initandlisten","msg":"Flow Control is
enabled on this deployment"}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.553+00:00"},"s":"I",
"c":"FTDC",     "id":20625,   "ctx":"initandlisten","msg":"Initializing full-
time diagnostic data
capture","attr":{"dataDirectory":"/data/db/diagnostic.data"}}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.562+00:00"},"s":"I",
"c":"NETWORK",  "id":23015,   "ctx":"listener","msg":"Listening
on","attr":{"address":"/tmp/mongodb-27017.sock"}}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.562+00:00"},"s":"I",
"c":"NETWORK",  "id":23015,   "ctx":"listener","msg":"Listening
on","attr":{"address":"0.0.0.0"}}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.562+00:00"},"s":"I",
"c":"NETWORK",  "id":23016,   "ctx":"listener","msg":"Waiting for
connections","attr":{"port":27017,"ssl":"off"}}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.681+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"LogicalSessionCacheRefresh","msg":"Slow
query","attr":{"type":"command","ns":"config.system.sessions","command":{"lis
tIndexes":"system.sessions","cursor":{},"$db":"config"},"numYields":0,"reslen
":245,"locks":{"ReplicationStateTransition":{"acquireCount":{"w":1}},"Global"
:{"acquireCount":{"r":1}},"Database":{"acquireCount":{"r":1}},"Collection":{"
acquireCount":{"r":1}},"Mutex":{"acquireCount":{"r":1}}},"storage":{},"protoc
ol":"op_msg","durationMillis":124}}
mongo            | {"t":{"$date":"2020-12-01T03:08:00.681+00:00"},"s":"I",
"c":"COMMAND",  "id":51803,   "ctx":"LogicalSessionCacheReap","msg":"Slow
query","attr":{"type":"command","ns":"config.system.sessions","command":{"lis
tIndexes":"system.sessions","cursor":{},"$db":"config"},"numYields":0,"reslen
":245,"locks":{"ReplicationStateTransition":{"acquireCount":{"w":1}},"Global"
:{"acquireCount":{"r":1}},"Database":{"acquireCount":{"r":1}},"Collection":{"
acquireCount":{"r":1}},"Mutex":{"acquireCount":{"r":1}}},"storage":{},"protoc
ol":"op_msg","durationMillis":124}}
mongo-express_1  | Tue Dec  1 03:08:01 UTC 2020 retrying to connect to
mongo:27017 (4/5)
mongo            | {"t":{"$date":"2020-12-01T03:08:01.412+00:00"},"s":"I",
"c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"172.22.0.2:60000","connectionId":1,"connectionCou
nt":1}}
mongo            | {"t":{"$date":"2020-12-01T03:08:01.592+00:00"},"s":"I",
"c":"NETWORK",  "id":22944,   "ctx":"conn1","msg":"Connection
ended","attr":{"remote":"172.22.0.2:60000","connectionId":1,"connectionCount"
:0}}
mongo-express_1  | Welcome to mongo-express
mongo-express_1  | ------------------------
mongo-express_1  |
```

```
mongo-express_1  |
mongo-express_1  | Mongo Express server listening at http://0.0.0.0:8081
mongo-express_1  | Server is open to allow connections from anyone (0.0.0.0)
mongo-express_1  | basicAuth credentials are "admin:pass", it is recommended
you change this in your config.js!
mongo            | {"t":{"$date":"2020-12-01T03:08:03.239+00:00"},"s":"I",
"c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"172.22.0.2:60002","connectionId":2,"connectionCou
nt":1}}
mongo            | {"t":{"$date":"2020-12-01T03:08:03.275+00:00"},"s":"I",
"c":"NETWORK",  "id":51800,   "ctx":"conn2","msg":"client
metadata","attr":{"remote":"172.22.0.2:60002","client":"conn2","doc":{"driver
":{"name":"nodejs","version":"2.2.24"},"os":{"type":"Linux","name":"linux","a
rchitecture":"x64","version":"4.19.104-microsoft-
standard"},"platform":"Node.js v12.20.0, LE, mongodb-core: 2.1.8"}}}
mongo-express_1  | Database connected
mongo            | {"t":{"$date":"2020-12-01T03:08:03.718+00:00"},"s":"I",
"c":"ACCESS",   "id":20250,   "ctx":"conn2","msg":"Successful
authentication","attr":{"mechanism":"SCRAM-SHA-
1","principalName":"root","authenticationDatabase":"admin","client":"172.22.0
.2:60002"}}
mongo-express_1  | Admin Database connected
mongo            | {"t":{"$date":"2020-12-01T03:08:03.748+00:00"},"s":"I",
"c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"172.22.0.2:60004","connectionId":3,"connectionCou
nt":2}}
mongo            | {"t":{"$date":"2020-12-01T03:08:03.858+00:00"},"s":"I",
"c":"ACCESS",   "id":20250,   "ctx":"conn3","msg":"Successful
authentication","attr":{"mechanism":"SCRAM-SHA-
1","principalName":"root","authenticationDatabase":"admin","client":"172.22.0
.2:60004"}}
mongo            | {"t":{"$date":"2020-12-01T03:08:03.861+00:00"},"s":"I",
"c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection
accepted","attr":{"remote":"172.22.0.2:60006","connectionId":4,"connectionCou
nt":3}}
mongo            | {"t":{"$date":"2020-12-01T03:08:03.908+00:00"},"s":"I",
"c":"ACCESS",   "id":20250,   "ctx":"conn4","msg":"Successful
authentication","attr":{"mechanism":"SCRAM-SHA-
1","principalName":"root","authenticationDatabase":"admin","client":"172.22.0
.2:60006"}}
```

check operation of stack at this stage

View the admin site in the browser at port 8081 and check some of the data noting the unique id values.



Author IDs

Using this information check the following testing links.

catalog/

catalog/books
catalog/authors
catalog/genres
catalog/bookinstances

catalog/book/:id
catalog/author/:id
catalog/genre/:id
catalog/bookinstance/:id

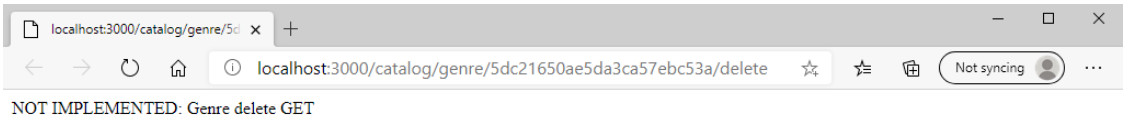
catalog/book/create
catalog/author/create
catalog/genre/create
catalog/bookinstance/create

catalog/book/:id/update
catalog/author/:id/update
catalog/genre/:id/update
catalog/bookinstance/:id/update

catalog/book/:id/delete
catalog/author/:id/delete

catalog/genre/:id/delete
catalog/bookinstance/:id/delete

For example



Router test
