Backend Bootcamp - Session 4: MongoDB Complete Overview

Agenda: MongoDB Introduction + Installation

- Introduction to MongoDB
- NoSQL vs SQL comparison
- MongoDB structure & terminology
- MongoDB installation on Windows
- Using MongoDB Compass

What is MongoDB?

- A NoSQL, document-oriented database
- Stores data in JSON-like documents (BSON)
- Schema-less and highly flexible
- Ideal for dynamic or hierarchical data

Why Use MongoDB?

- Easily scalable and distributed
- Schema can evolve over time (no ALTER TABLE needed)
- Works well with JavaScript and Node.js
- Great for rapid prototyping and agile development

MongoDB vs SQL (Comparison)

- SQL uses tables, rows, and columns
- MongoDB uses collections and documents
- SQL is schema-based; MongoDB is schema-less
- SQL uses JOINs; MongoDB supports embedding/nested data

MongoDB Structure Overview

- **Database**: Holds collections
- **Collection**: Like a table; holds many documents
- **Document**: A JSON-like object
- **Field**: Key-value pair inside a document

Document Example

```json
```

```
{
  "name": "Ayesha",
  "age": 22,
  "skills": ["Node.js", "MongoDB"],
  "isStudent": true
}
```

Installing MongoDB on Windows (Step-by-Step)

- Go to: https://mongodb.com/try/download/community
- Select:
  - Version: Current release
  - Platform: Windows
  - Package: .msi installer
- Run installer and choose 'Complete' setup

Configuration During Install

- Choose default service configuration (Network Service)
- Keep default data directory
- Ensure Compass is checked if you want GUI
- Click Install and wait for setup to complete

Add MongoDB to System PATH

- Open 'Environment Variables' in system settings
- Edit the 'Path' variable
- Add: C:\Program Files\MongoDB\Server\<version>\bin
- This lets you run mongod and mongosh from any terminal

Verify Installation

- Open Command Prompt and type:
  - mongod --version
  - mongosh
- If you see shell or version info, installation was successful

Install MongoDB Compass

- MongoDB Compass is a GUI to view and manage your data
- If not installed with MongoDB:

  - Go to https://mongodb.com/try/download/compass
  - Download Windows installer
  - Run and install

Using MongoDB Compass

- Open Compass and connect to:
  mongodb://localhost:27017
- View databases, collections, and documents
- Use built-in visual query builder and data browser

Agenda: MongoDB Fundamentals

- Introduction to MongoDB and NoSQL
- Setting up MongoDB locally and via the cloud
- Performing CRUD operations: Create, Read, Update, Delete

What is MongoDB?

- A NoSQL database that stores data in flexible, JSON-like documents
- Instead of rows and columns, data is stored in collections of documents
- Schema-less: each document can have a different structure

Why NoSQL over SQL?

- SQL (e.g., MySQL, PostgreSQL) uses structured tables with schemas
- NoSQL (e.g., MongoDB) stores data in documents, allowing flexibility
- Ideal for rapidly changing data, nested fields, and large-scale applications

MongoDB Terminology

- **Database**: Container for collections
- **Collection**: Like a table in SQL, holds multiple documents
- **Document**: Core data structure, like a JSON object (stored as BSON)
- **Field**: Key-value pair inside a document

Installing MongoDB

- Download from mongodb.com/try/download/community
- Use MongoDB Compass for GUI, or `mongosh` for CLI
- Start MongoDB server using `mongod`
- Optionally use MongoDB Atlas (cloud DB) with free tier

Connecting MongoDB to Node.js

- Install native driver: `yarn add mongodb`

- Basic connection code:

```js
const { MongoClient } = require('mongodb');
const uri = 'mongodb://localhost:27017';
const client = new MongoClient(uri);
await client.connect();
```

CREATE: Insert Documents

- Insert a single document:

```js
db.users.insertOne({ name: 'Ali', age: 25 });
```

- Insert multiple documents:

```js
db.users.insertMany([{ name: 'Sara' }, { name: 'Omar' }]);
```

READ: Querying Documents

- Find all users:

```js
db.users.find();
```

- Find with a filter:

```js
db.users.find({ age: { $gte: 18 } });
```

- Find one:

```js
db.users.findOne({ name: 'Ali' });
```

UPDATE: Modify Documents

- Update a single document:

```js
db.users.updateOne({ name: 'Ali' }, { $set: { age: 30 } });
```

- Update multiple documents:

```js
db.users.updateMany({}, { $set: { active: true } });
```

DELETE: Remove Documents

- Delete one:

```js
db.users.deleteOne({ name: 'Ali' });
```

- Delete many:

```js
db.users.deleteMany({ active: false });
```

Practice Exercise (Optional)

- Create a `products` collection with documents containing `name`, `price`, and `category`
- Perform:
  1. Insertion of multiple product records
  2. Query products above a certain price
  3. Update product category
  4. Delete discontinued products

Bonus Tips

- MongoDB uses BSON internally (Binary JSON) for efficiency
- Fields can be deeply nested or arrays
- Use indexes to speed up large queries
- Use Compass to visually explore documents and collections