Zaawansowane Programowanie Webowe

Lab 7. Socket.IO i MongoDB

Cel zajęć:

Poznanie biblioteki Socket.IO oraz bazy MongoDB i użycie ich w aplikacji webowej.

Narzędzia:
Node.js (https://nodejs.org/) + NPM + edytor kodu
lub w przeglądarce https://codesandbox.io/

Socket.IO is a JavaScript library for realtime web applications. [12]

- It enables realtime, bi-directional communication between web clients and servers.
- Like Node.js, it is event-driven.
- It has two parts:
  - a client-side library that runs in the browser,
  - a server-side library for Node.js.

  Both components have a nearly identical API.

Socket.IO primarily uses the WebSocket protocol with polling as a fallback option, while providing the same interface. Although it can be used as simply a wrapper for WebSocket, it provides many more features, including broadcasting to multiple sockets, storing data associated with each client, and asynchronous I/O.

It can be installed with npm. `npm install socket.io`

Example:

```
const server = require('http').createServer();
const io = require('socket.io')(server);
io.on('connection', client => {
  client.on('event', data => {
    /* do sth */
  });
  client.on('disconnect', () => {
    /* do sth */
  });
});
server.listen(3000);
```

[12] https://en.wikipedia.org/wiki/Socket.IO

Cheetsheat on Socket.IO methods [13]

https://socket.io/docs/emit-cheatsheet/

```
  // sending to sender-client only
  socket.emit('message', "this is a test");

  // sending to all clients, include sender
  io.emit('message', "this is a test");

  // sending to all clients except sender
  socket.broadcast.emit('message', "this is a test");

  // sending to all clients in 'game' room(channel) except sender
  socket.broadcast.to('game').emit('message', 'nice game');

  // sending to all clients in 'game' room(channel), include sender
  io.in('game').emit('message', 'cool game');

  // sending to sender client, only if they are in 'game' room(channel)
  socket.to('game').emit('message', 'enjoy the game');

  // sending to all clients in namespace 'myNamespace', include sender
  io.of('myNamespace').emit('message', 'gg');

  // sending to individual socketid (server-side)
  socket.broadcast.to(socketid).emit('message', 'for your eyes only');

  // join to subscribe the socket to a given channel (server-side):
  socket.join('some room');

  // then simply use to or in (they are the same) when broadcasting or emitting
(server-side)
  io.to('some room').emit('some event'):

  // leave to unsubscribe the socket to a given channel (server-side)
  socket.leave('some room');
```

MongoDB[14]

MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema.
MongoDB provides high performance, high availability, and easy scalability.

Database [15]

Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

---

[13] https://gist.github.com/alexpchin/3f257d0bb813e2c8c476
[14] https://www.mongodb.com/
[15] https://www.tutorialspoint.com/mongodb/mongodb_overview.htm

Collection

Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

Document

A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

RDBMS vs. MongoDB	
Database	Database
Table	Collection
Tuple/Row	Document
Column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by mongodb itself)

Sample Document

```
{
   _id: ObjectId(7df78ad8902c)
   title: 'MongoDB Overview',
   description: 'MongoDB is no sql database',
   by: 'tutorials point',
   url: 'http://www.tutorialspoint.com',
   tags: ['mongodb', 'database', 'NoSQL'],
   likes: 100,
   comments: [
      {
         user:'user1',
         message: 'My first comment',
         dateCreated: new Date(2011,1,20,2,15),
         like: 0
      },
      {
         user:'user2',
         message: 'My second comments',
         dateCreated: new Date(2011,1,25,7,45),
         like: 5
      }
   ]
}
```

_id is a 12 bytes hexadecimal number which assures the uniqueness of every document. You can provide _id while inserting the document. If you don't provide then MongoDB provides a unique id for every document.

More: https://www.tutorialspoint.com/mongodb/index.htm

Mongoose
Mongoose is a MongoDB object modeling tool for Node.js. https://mongoosejs.com/
`npm install mongoose`

Example of usage https://mongoosejs.com/docs/index.html

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

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

const Cat = mongoose.model('Cat', { name: String });

const kitty = new Cat({ name: 'Zildjian' });
kitty.save().then(() => console.log('meow'));
```

Zadanie

Przejdź tutorial i zaimplementuj chat używając Node'a oraz Socket.io. Link do tutorialu: https://medium.com/@noufel.gouirhate/build-a-simple-chat-app-with-node-js-and-socket-io-ea716c093088
Uzupełnij kod o przechowywanie historii chatu w bazie MongoDB.
Użyj Mongoose oraz mLab Sandbox- https://mlab.com/ - Database-as-a-Service for MongoDB.
[10pkt]