Assignment 3 (6 points total)

In this assignment we will get familiar with the basic principles of databases and take a look at UART which we'll use to connect our IoT device and our web server.

Prerequisites:

- Have Node and Express (with cookie and body parsers) installed
- Install `serialport`, `mongodb` and `sqlite3` NPM packages
- Install MongoDB server on your computer

Tasks 1 and 2 (4 points total)

In this task you will have to create an Express application for working with a car dealer stock (much like in assignment 2 task 1), but use a database for storing the car data. Your application should at least have the following routes implemented:

- `GET /` - Returns a list of all cars available
- `POST /` - Adds a car to the list
- `GET /:carID` - Returns a car with the specified ID
- `GET /makes/:make` - Returns all cars of a specified make
- `DELETE /:carID` - Deletes a car with the specified ID

The list of the cars should persist when the server is restarted. You must generate an ID on the server when adding a new car to the list. If you wish, you can have additional routes, for example modifying a car or retrieving cars by production year, etc.

Store the car data in a **MongoDB database for task 1** and an **SQLite database for task 2**. Otherwise tasks 1 and 2 are identical.
Hints:

- You can use callbacks, Promises or async/await for asynchronous calls
- `util.promisify` can turn callback-based Node functions into Promise-based versions
- Use `body-parser` to retrieve data from the request body

Task 3 (2 points)

In this task you will be required to use Node serialport library to communicate with your microcontroller dev kit and signal capture board. We will be re-using our Dice webpage from Assignment 1 task 3. The difference here is that your roll button should trigger your microcontroller to generate a new value for the die through UART. The resulting value should be returned through UART to your server and ultimately display the correct face of the die in your browser. **You will be required to procure a signal capture board and an LPCxpresso from the Metropolia library.**
https://metropolia.finna.fi/Record/3amk.286179 &
https://metropolia.finna.fi/Record/3amk.139122

You will send 'a' followed by 'q' to the µC over UART to get two strings back(also over UART)The second string will contain the die value followed by a space and then a string of bits

Example: Send(button down) 'a' get back "0 0000000" send(button up) 'q' receive "5 1011101"

In the above example 5 is the value shown on the die face image that you should serve to the client.

Hints:

- The serial port library can be installed using ‘npm’
- The serialport library uses node.js stream interface
- You can use events, but must use caution as event listeners can cause some unforeseen problems. As such, you must manage when you listen and when you don’t (manually).
- This was completed without having to change the original HTLM or EJS files at all. All changes were carried out in the backend.

IMPORTANT INFORMATION

Demonstrate your working assignments to the teacher in class before the deadline for full points. Assignments submitted even a moment after the deadline will only be able to get half points.
!!!There is no exception to this rule!!!