Exercise (Instructions): Node and MongoDB Part 2

Objectives and Outcomes

In this exercise you will continue to explore communicating from your Node application to the MongoDB server. At the end of this exercise you will be able to:

- Develop a Node module containing some common MongoDB operations
- Use the Node module in your application and communicate with the MongoDB server

Implementing a Node Module of Database Operations

- Create a new file named *operations.js* that contains a few MongoDB operations and add the following code:

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

exports.insertDocument = (db, document, collection, callback) => {
    const coll = db.collection(collection);
    coll.insert(document, (err, result) => {
        assert.equal(err, null);
        console.log("Inserted " + result.result.n +
            " documents into the collection " + collection);
        callback(result);
    });
};

exports.findDocuments = (db, collection, callback) => {
    const coll = db.collection(collection);
    coll.find({}).toArray((err, docs) => {
        assert.equal(err, null);
        callback(docs);
    });
};

exports.removeDocument = (db, document, collection, callback) => {
    const coll = db.collection(collection);
    coll.deleteOne(document, (err, result) => {
        assert.equal(err, null);
        console.log("Removed the document ", document);
        callback(result);
    });
};

exports.updateDocument = (db, document, update, collection, callback) => {
    const coll = db.collection(collection);
    coll.updateOne(document, { $set: update }, null, (err, result) => {
        assert.equal(err, null);
        console.log("Updated the document with ", update);
        callback(result);
    });
};

```

Using the Node Module for Database Operations

- Update the file named *index.js* as follows:

```
. . .

const dboper = require('./operations');

. . .

    dboper.insertDocument(db, { name: "Vadonut", description: "Test"},
        "dishes", (result) => {
            console.log("Insert Document:\n", result.ops);

            dboper.findDocuments(db, "dishes", (docs) => {
                console.log("Found Documents:\n", docs);

                dboper.updateDocument(db, { name: "Vadonut" },
                    { description: "Updated Test" }, "dishes",
                    (result) => {
                        console.log("Updated Document:\n", result.result);

                        dboper.findDocuments(db, "dishes", (docs) => {
                            console.log("Found Updated Documents:\n", docs);

                            db.dropCollection("dishes", (result) => {
                                console.log("Dropped Collection: ", result);

                                client.close();
                            });
                        });
                    });
            });
    });

. . .
```

- Run the server by typing the following at the prompt and observe the results:

```
npm start
```

- Do a Git commit with the message "Node MongoDB Example 2".

Conclusions

In this exercise you created a Node module to package some database operations, and then used the module to interact with the MongoDB server.

✓ Complete