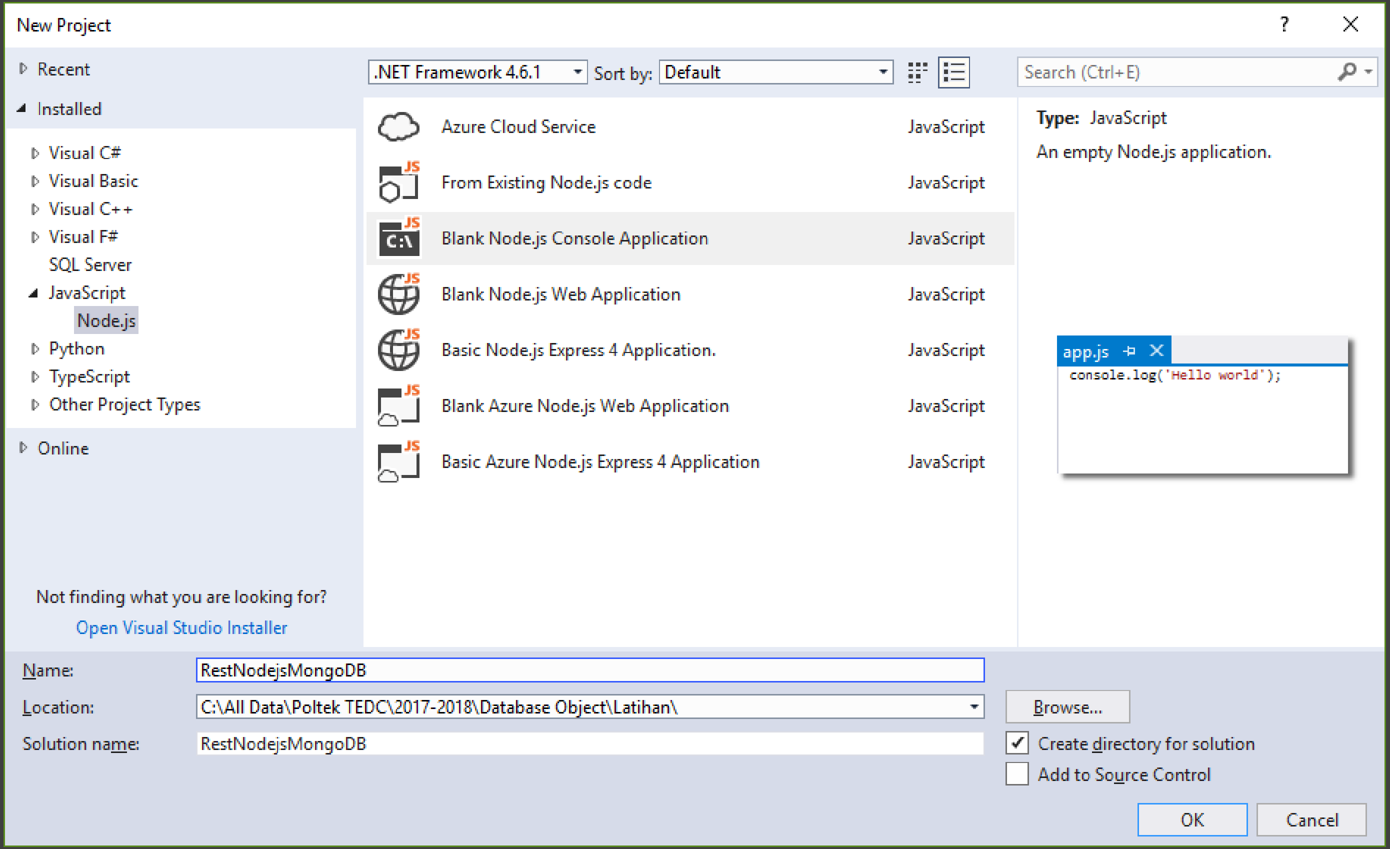
Restful API
NodeJS and MongoDB

Project Baru


New Project

Type: JavaScript

An empty Node.js application.

Azure Cloud Service	JavaScript
From Existing Node.js code	JavaScript
Blank Node.js Console Application	JavaScript
Blank Node.js Web Application	JavaScript
Basic Node.js Express 4 Application.	JavaScript
Blank Azure Node.js Web Application	JavaScript
Basic Azure Node.js Express 4 Application	JavaScript

```
console.log('Hello world');
```

Not finding what you are looking for?

Open Visual Studio Installer

Name: RestNodejsMongoDB

Location: C:\All Data\Poltek TEDC\2017-2018\Database Object\Latihan\

Solution name: RestNodejsMongoDB

[✓] Create directory for solution

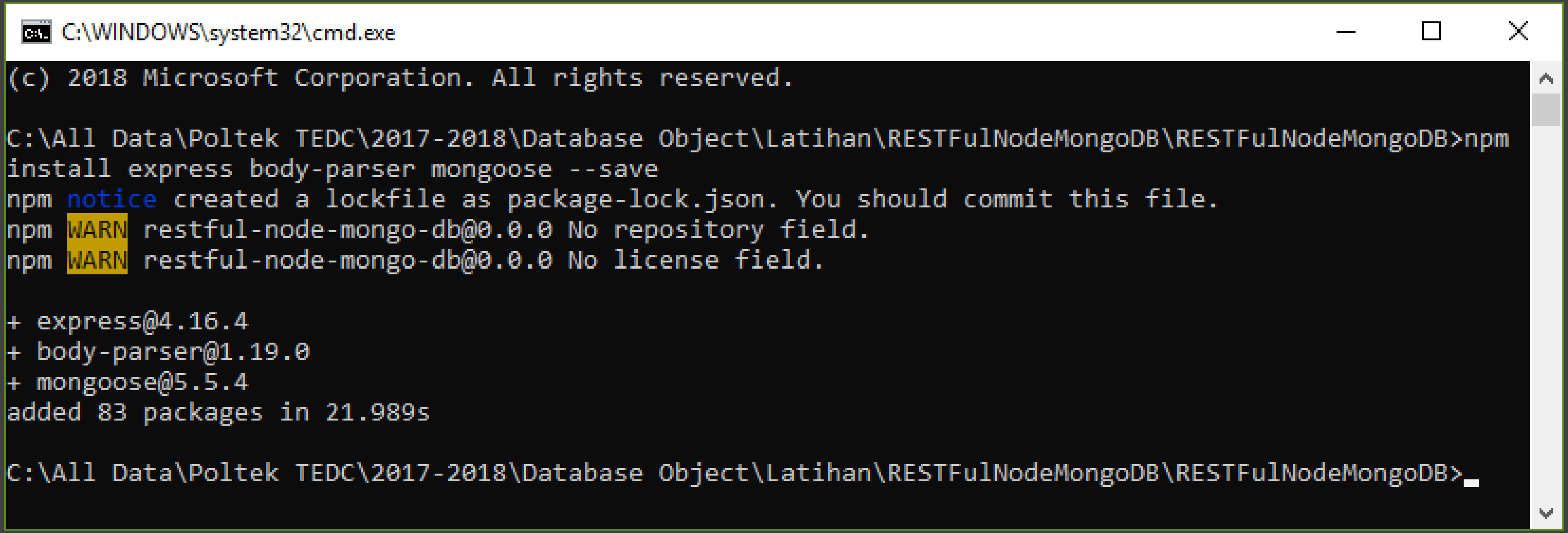
[ ] Add to Source Control

Import Lybrary

Pada solution explorer, klik kanan, pilih “Open Command Prompt From Here”

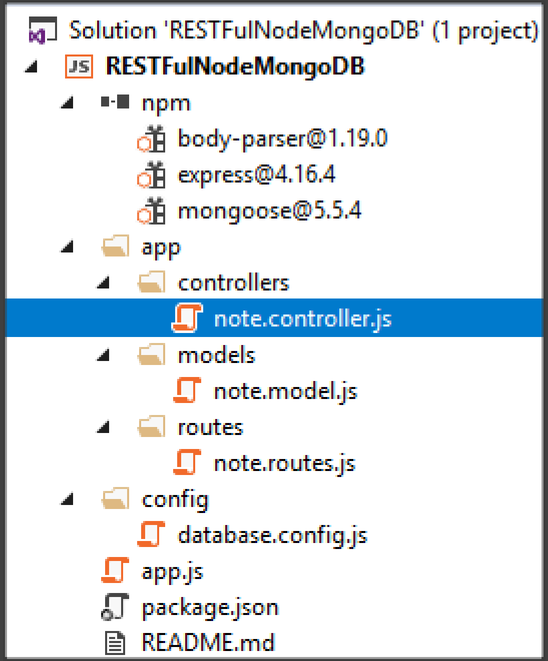
Ketikkan perintah :

- npm install --save body-parser express mongoose



```
(c) 2018 Microsoft Corporation. All rights reserved.

C:\All Data\Poltek TEDC\2017-2018\Database Object\Latihan\RESTFulNodeMongoDB\RESTFulNodeMongoDB>npm
install express body-parser mongoose --save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN restful-node-mongo-db@0.0.0 No repository field.
npm WARN restful-node-mongo-db@0.0.0 No license field.

+ express@4.16.4
+ body-parser@1.19.0
+ mongoose@5.5.4
added 83 packages in 21.989s

C:\All Data\Poltek TEDC\2017-2018\Database Object\Latihan\RESTFulNodeMongoDB\RESTFulNodeMongoDB>
```

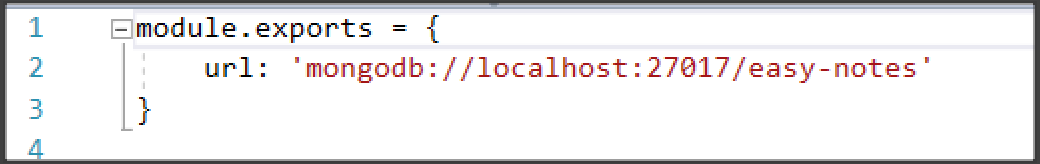
Hierarki Project Folder



```
Solution 'RESTFulNodeMongoDB' (1 project)
◢ RESTFulNodeMongoDB
   ◢ npm
        body-parser@1.19.0
        express@4.16.4
        mongoose@5.5.4
   ◢ app
      ◢ controllers
           note.controller.js
      ◢ models
           note.model.js
      ◢ routes
           note.routes.js
   ◢ config
        database.config.js
     app.js
     package.json
     README.md
```

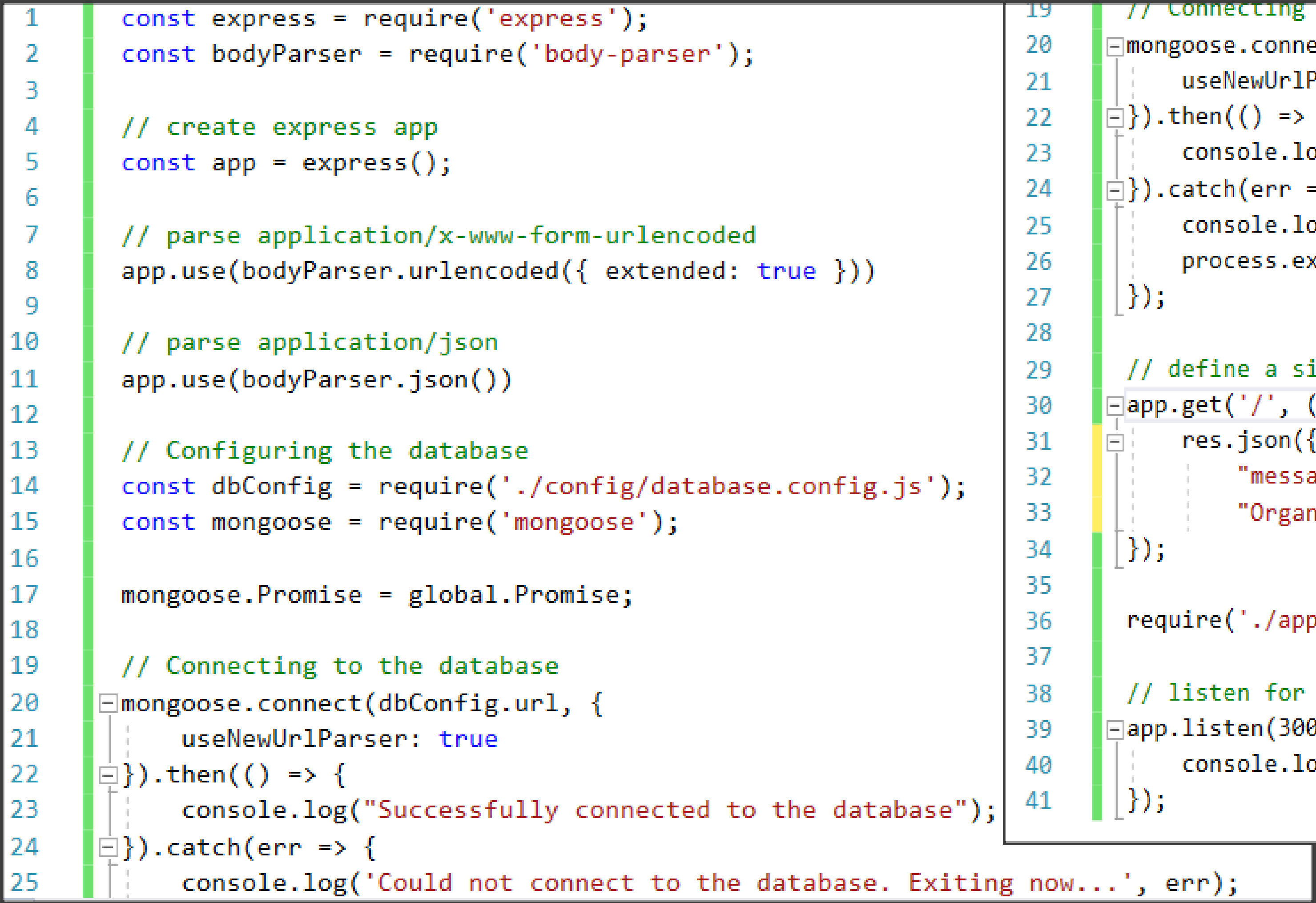
Config

Config -> database.config.js

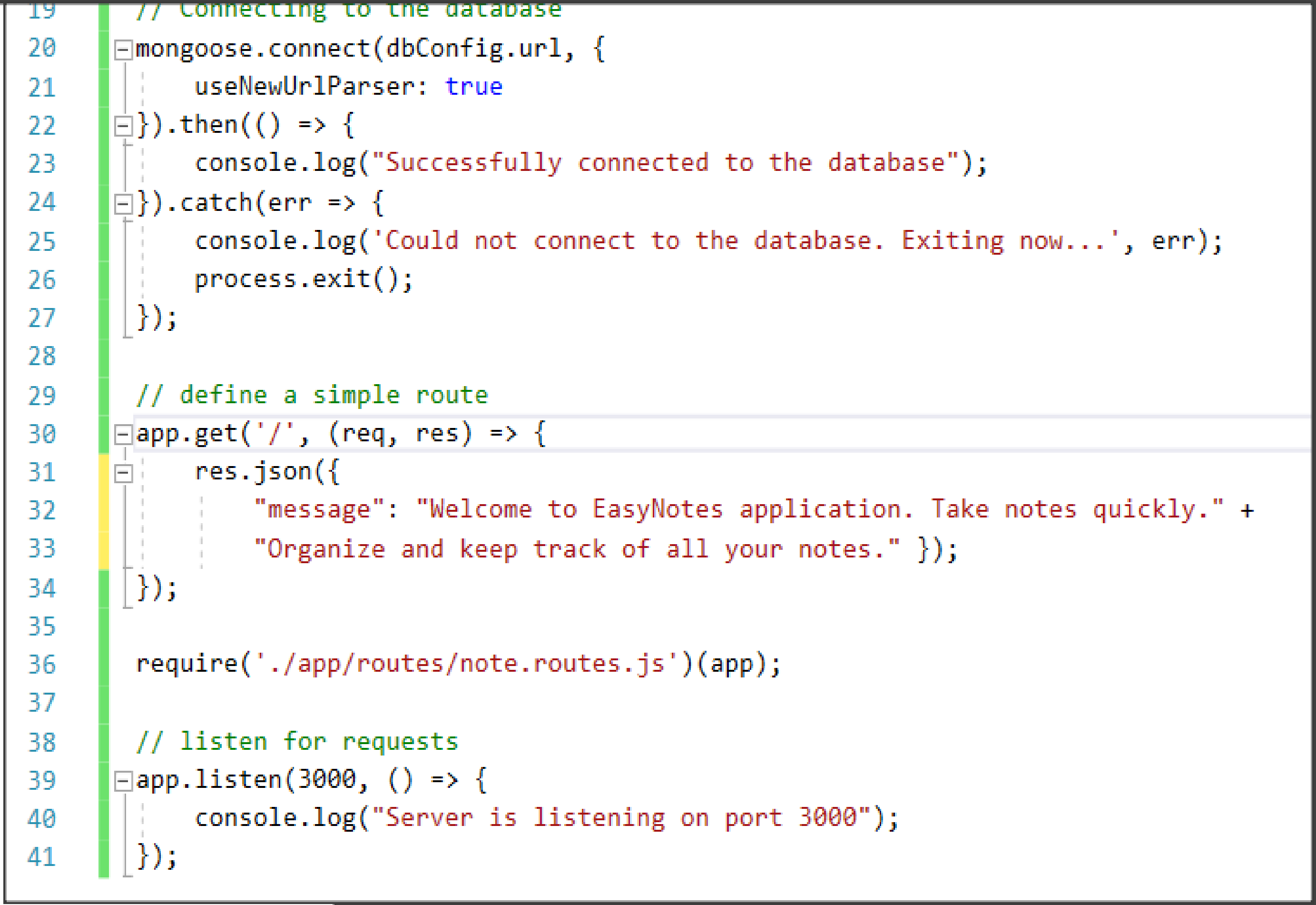


```
module.exports = {
    url: 'mongodb://localhost:27017/easy-notes'
}
```

app.js



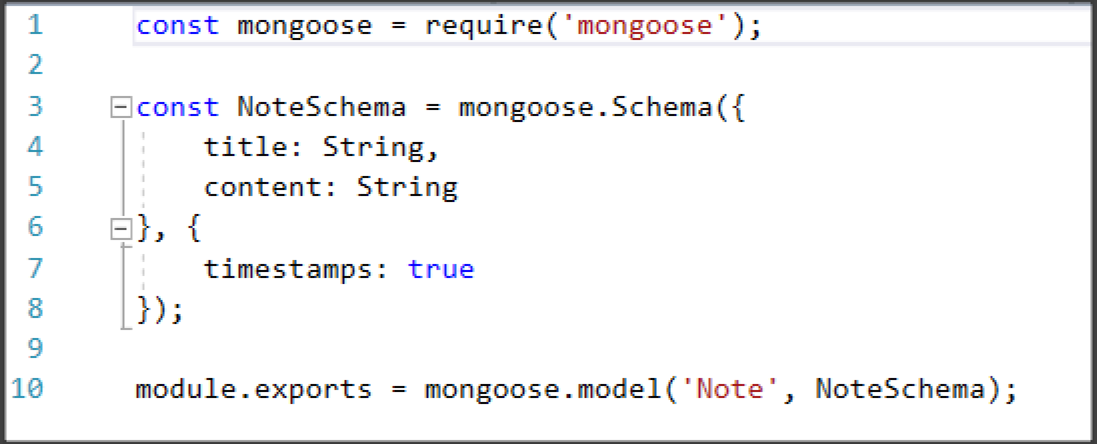
```javascript
const express = require('express');
const bodyParser = require('body-parser');

// create express app
const app = express();

// parse application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: true }))

// parse application/json
app.use(bodyParser.json())

// Configuring the database
const dbConfig = require('./config/database.config.js');
const mongoose = require('mongoose');

mongoose.Promise = global.Promise;

// Connecting to the database
mongoose.connect(dbConfig.url, {
    useNewUrlParser: true
}).then(() => {
    console.log("Successfully connected to the database");
}).catch(err => {
    console.log('Could not connect to the database. Exiting now...', err);
```



```javascript
// Connecting to the database
mongoose.connect(dbConfig.url, {
    useNewUrlParser: true
}).then(() => {
    console.log("Successfully connected to the database");
}).catch(err => {
    console.log('Could not connect to the database. Exiting now...', err);
    process.exit();
});

// define a simple route
app.get('/', (req, res) => {
    res.json({
        "message": "Welcome to EasyNotes application. Take notes quickly." +
        "Organize and keep track of all your notes." });
});

require('./app/routes/note.routes.js')(app);

// listen for requests
app.listen(3000, () => {
    console.log("Server is listening on port 3000");
});
```

Models

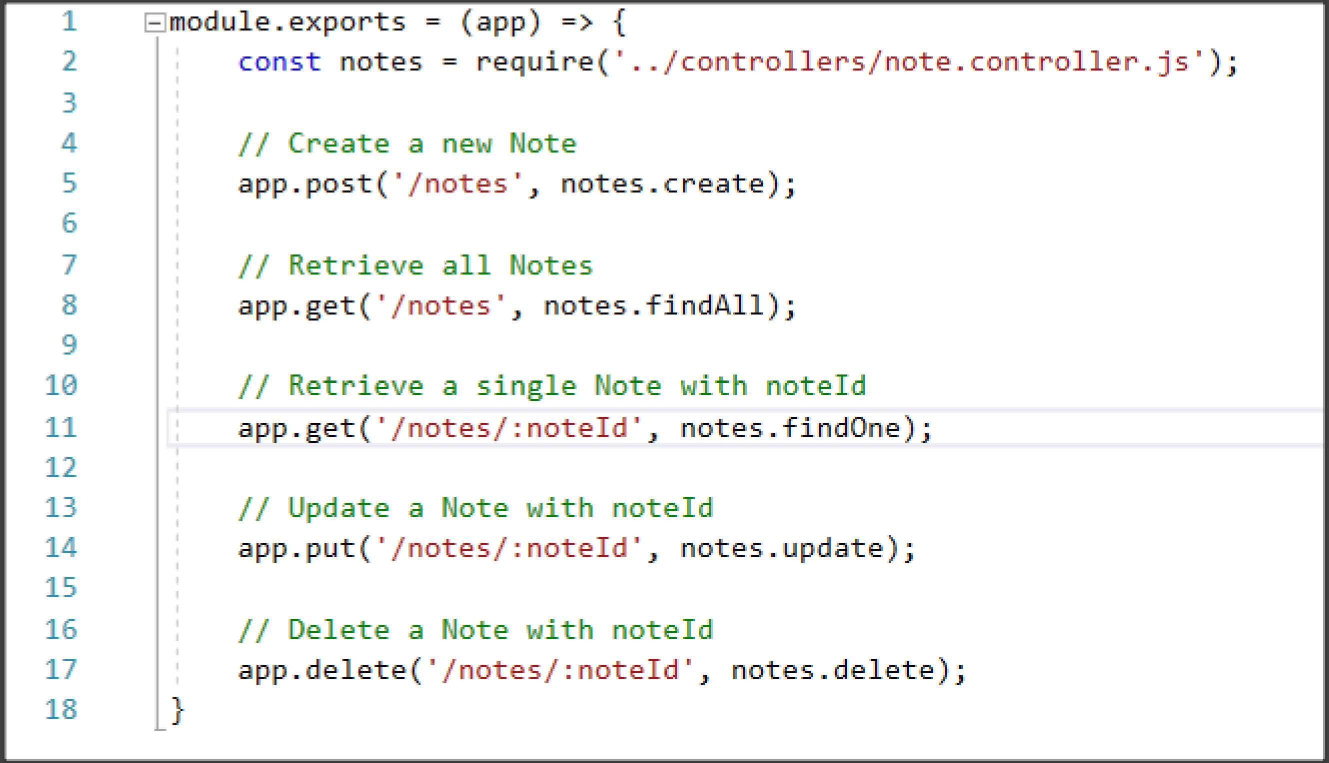
App.models -> note.models.js



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

const NoteSchema = mongoose.Schema({
    title: String,
    content: String
}, {
    timestamps: true
});

module.exports = mongoose.model('Note', NoteSchema);
```

Routes

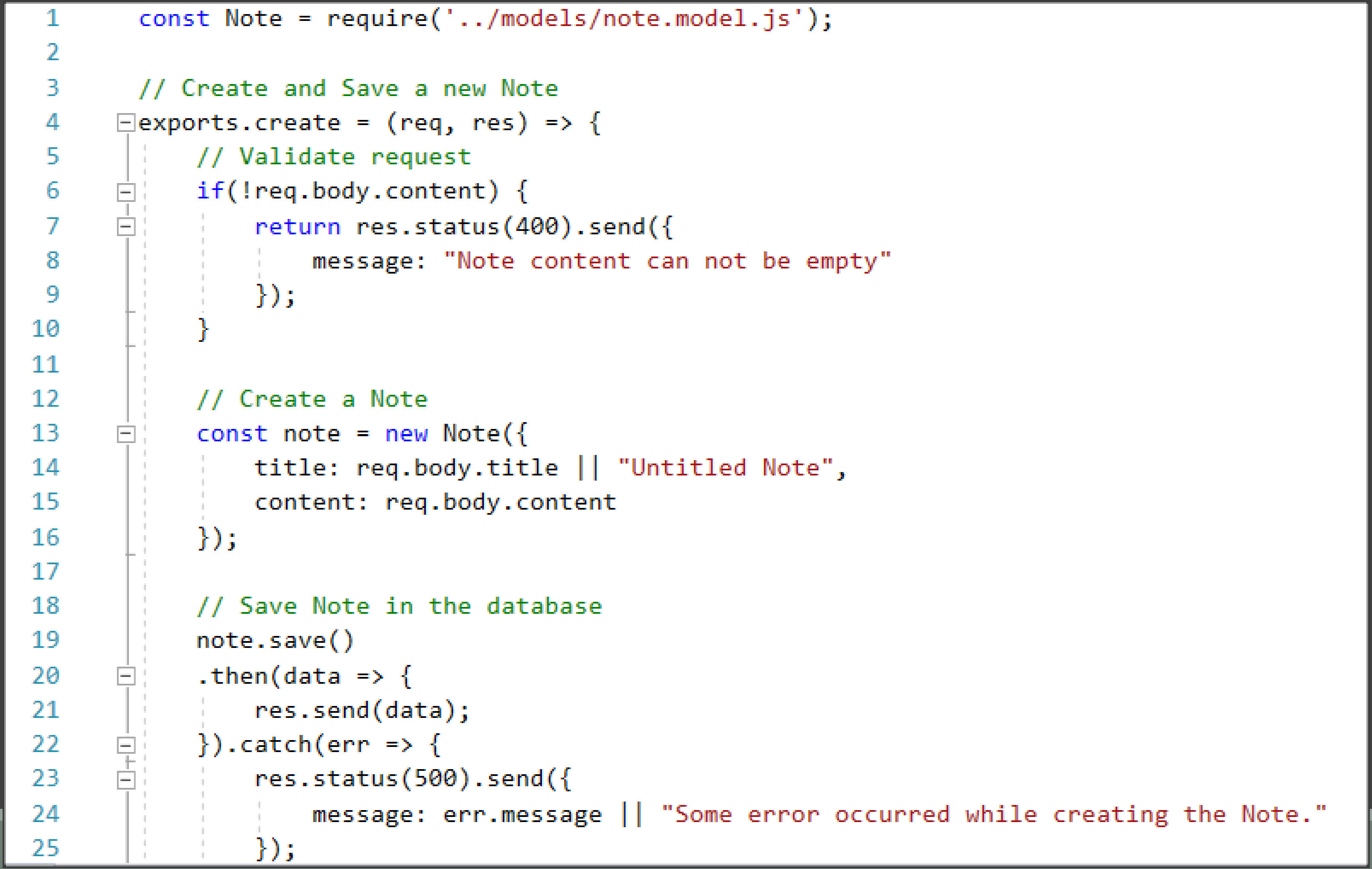
App.routes -> note.routes.js



```
module.exports = (app) => {
    const notes = require('../controllers/note.controller.js');

    // Create a new Note
    app.post('/notes', notes.create);

    // Retrieve all Notes
    app.get('/notes', notes.findAll);

    // Retrieve a single Note with noteId
    app.get('/notes/:noteId', notes.findOne);

    // Update a Note with noteId
    app.put('/notes/:noteId', notes.update);

    // Delete a Note with noteId
    app.delete('/notes/:noteId', notes.delete);
}
```

Controllers (1)

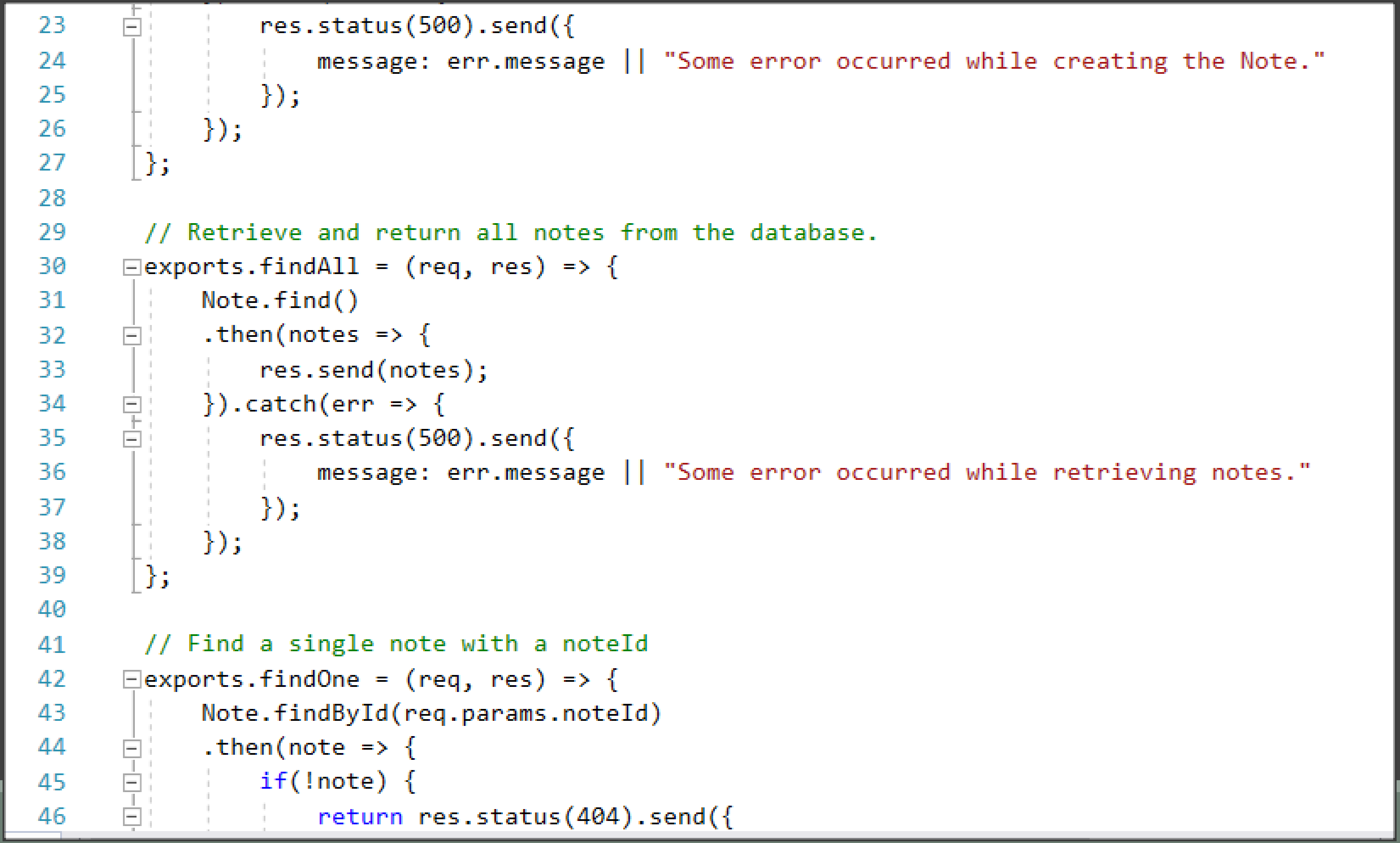
App.controllers -> note.controller.js



```
const Note = require('../models/note.model.js');

// Create and Save a new Note
exports.create = (req, res) => {
    // Validate request
    if(!req.body.content) {
        return res.status(400).send({
            message: "Note content can not be empty"
        });
    }

    // Create a Note
    const note = new Note({
        title: req.body.title || "Untitled Note",
        content: req.body.content
    });

    // Save Note in the database
    note.save()
    .then(data => {
        res.send(data);
    }).catch(err => {
        res.status(500).send({
            message: err.message || "Some error occurred while creating the Note."
        });
```

Controllers (2)

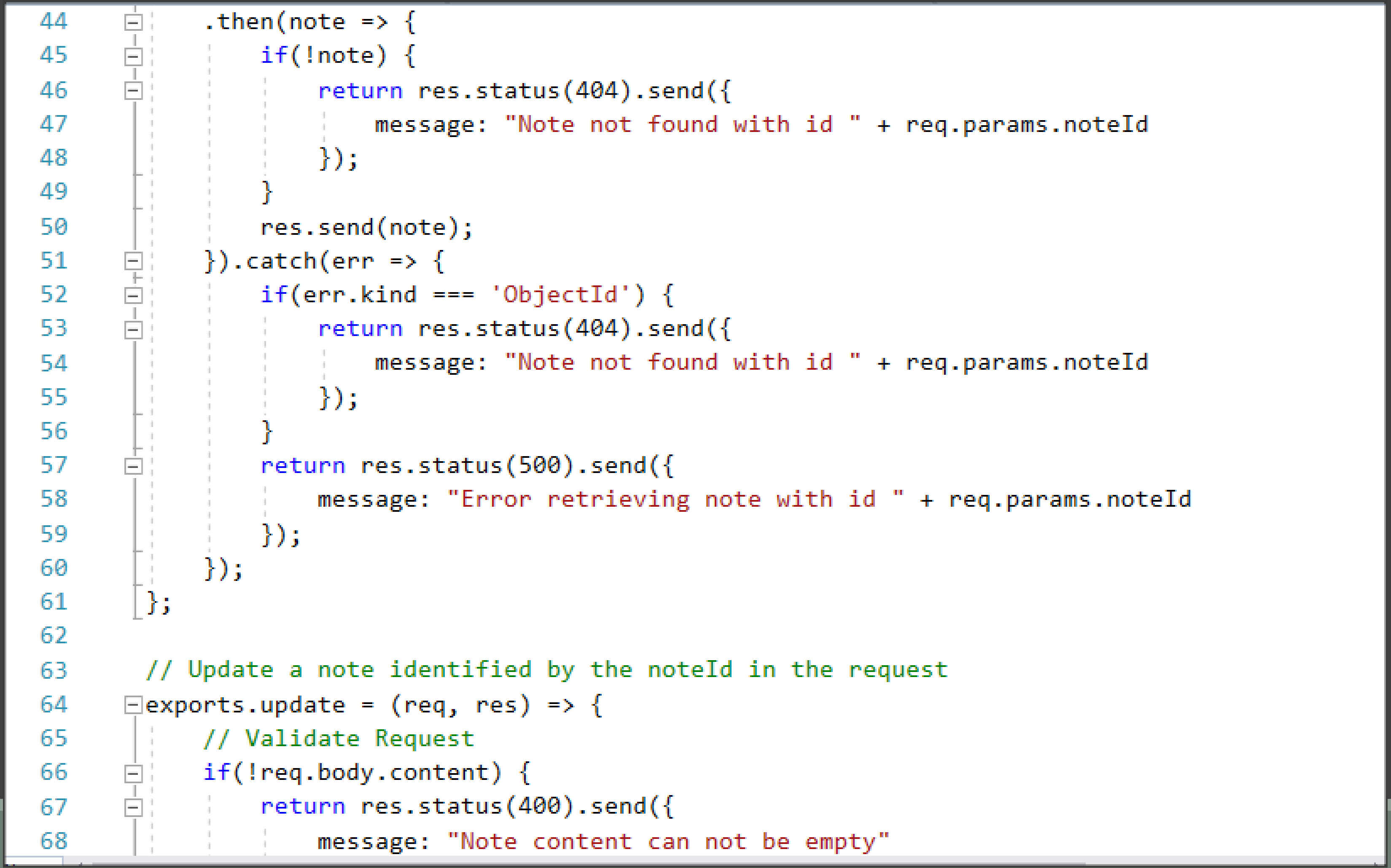
App.controllers -> note.controller.js



```
        res.status(500).send({
            message: err.message || "Some error occurred while creating the Note."
        });
    });
};

// Retrieve and return all notes from the database.
exports.findAll = (req, res) => {
    Note.find()
    .then(notes => {
        res.send(notes);
    }).catch(err => {
        res.status(500).send({
            message: err.message || "Some error occurred while retrieving notes."
        });
    });
};

// Find a single note with a noteId
exports.findOne = (req, res) => {
    Note.findById(req.params.noteId)
    .then(note => {
        if(!note) {
            return res.status(404).send({
```

Controllers (3)

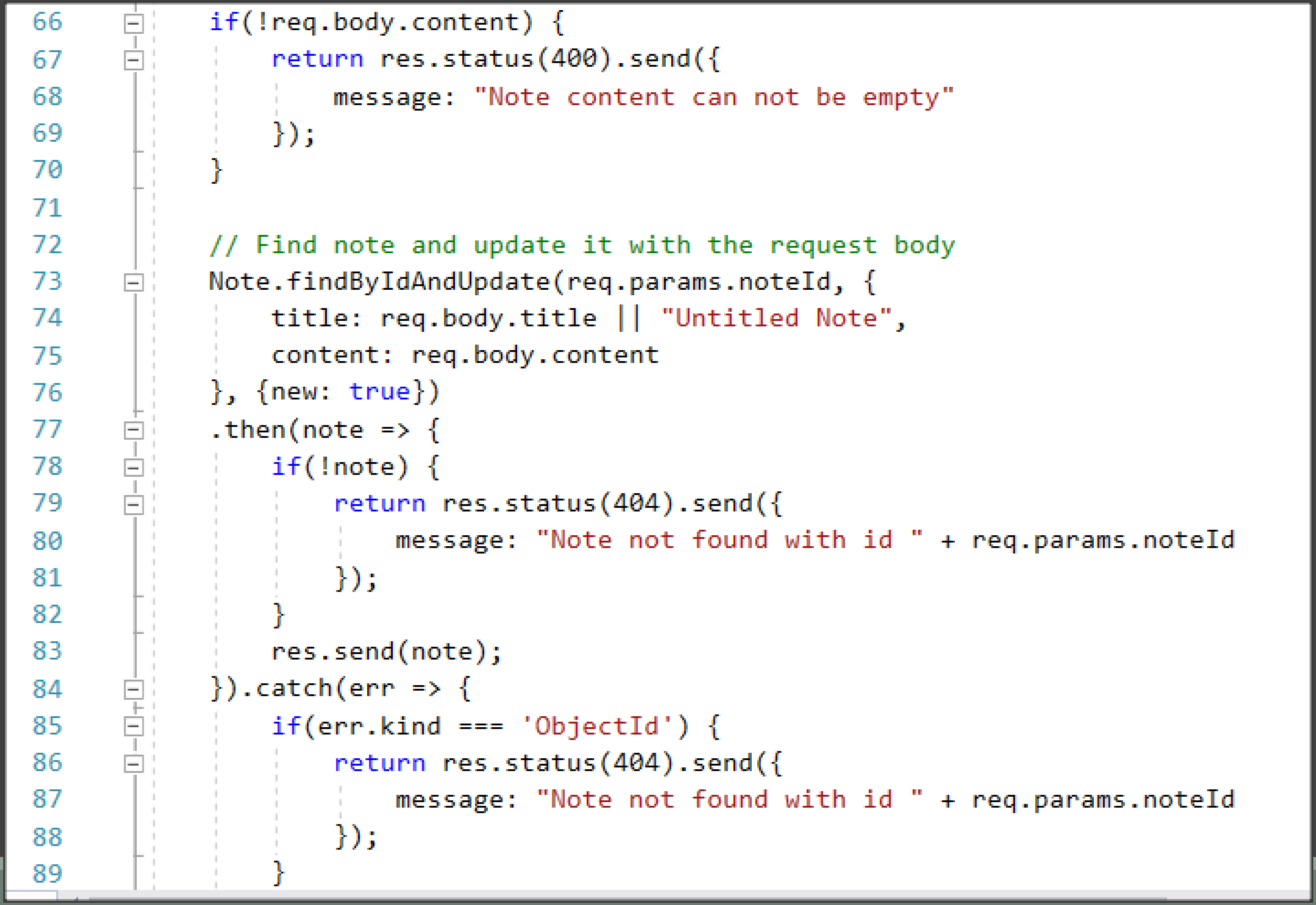
App.controllers -> note.controller.js



```
        .then(note => {
            if(!note) {
                return res.status(404).send({
                    message: "Note not found with id " + req.params.noteId
                });
            }
            res.send(note);
        }).catch(err => {
            if(err.kind === 'ObjectId') {
                return res.status(404).send({
                    message: "Note not found with id " + req.params.noteId
                });
            }
            return res.status(500).send({
                message: "Error retrieving note with id " + req.params.noteId
            });
        });
};

// Update a note identified by the noteId in the request
exports.update = (req, res) => {
    // Validate Request
    if(!req.body.content) {
        return res.status(400).send({
            message: "Note content can not be empty"
```

Controllers (4)

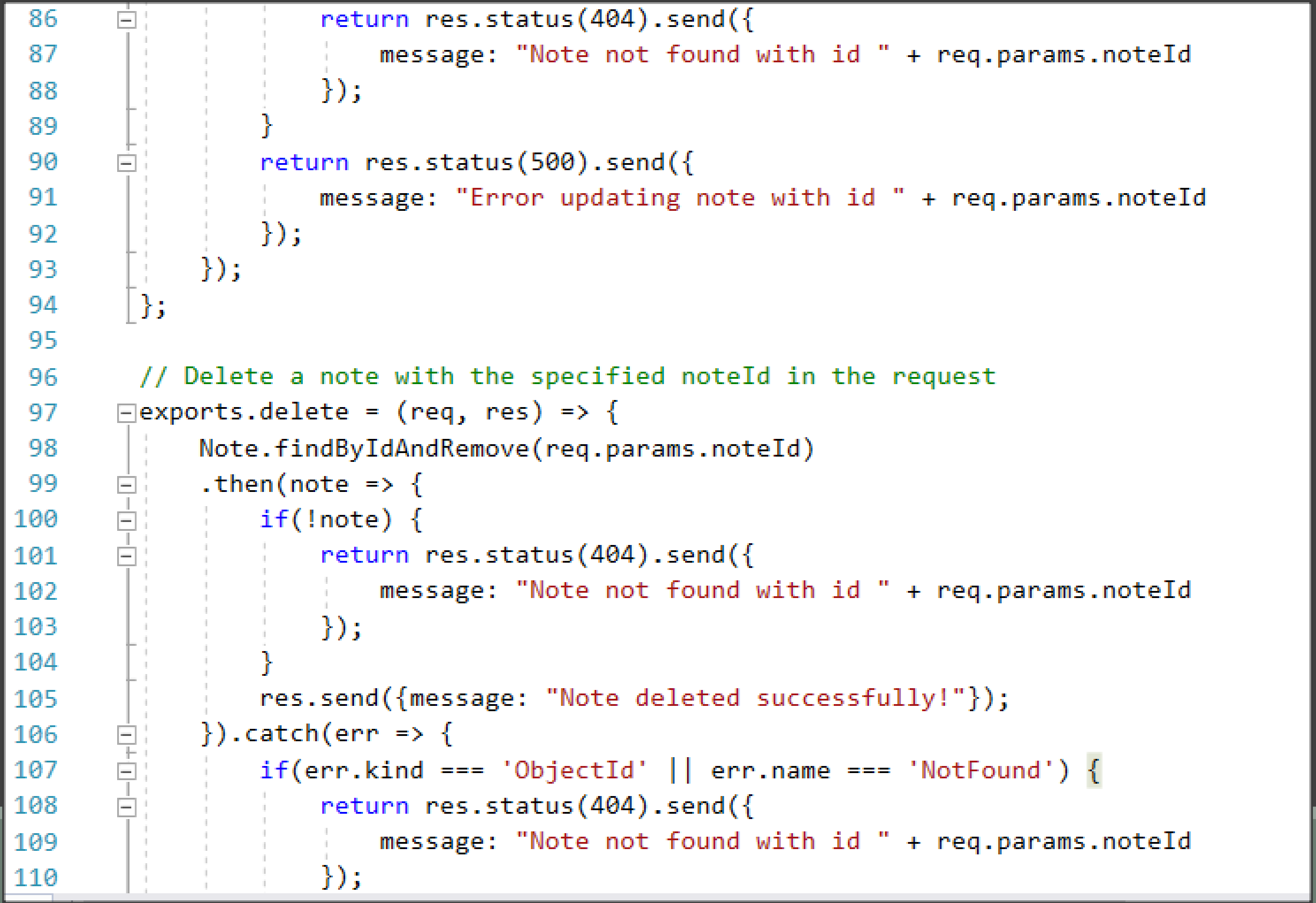
App.controllers -> note.controller.js



```
    if(!req.body.content) {
        return res.status(400).send({
            message: "Note content can not be empty"
        });
    }

    // Find note and update it with the request body
    Note.findByIdAndUpdate(req.params.noteId, {
        title: req.body.title || "Untitled Note",
        content: req.body.content
    }, {new: true})
    .then(note => {
        if(!note) {
            return res.status(404).send({
                message: "Note not found with id " + req.params.noteId
            });
        }
        res.send(note);
    }).catch(err => {
        if(err.kind === 'ObjectId') {
            return res.status(404).send({
                message: "Note not found with id " + req.params.noteId
            });
        }
```

Controllers (5)

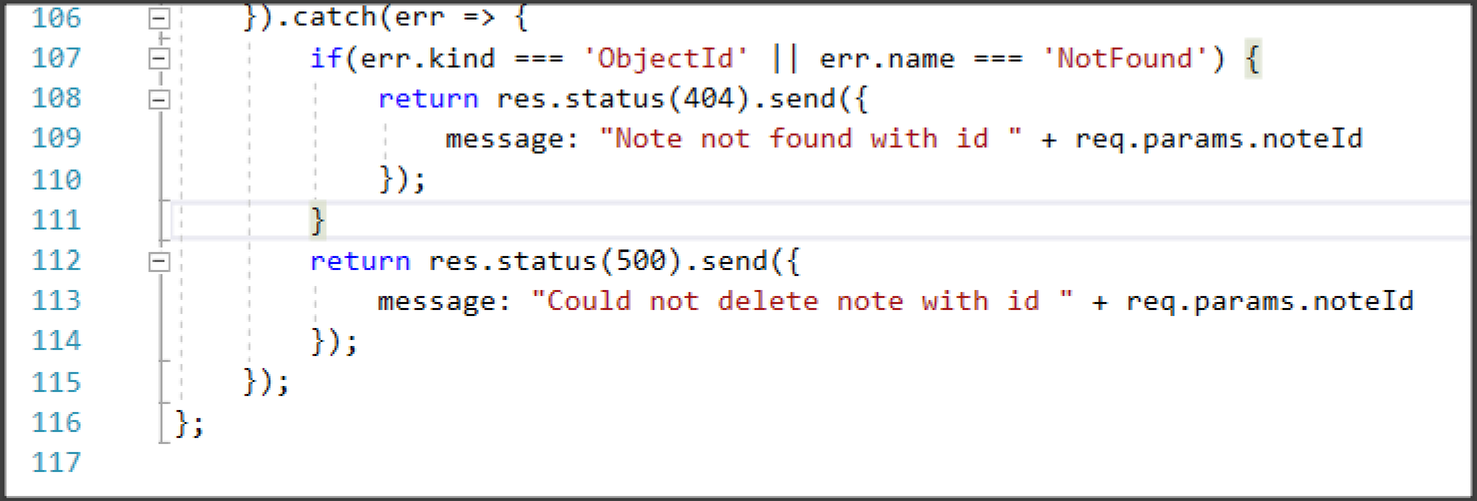
App.controllers -> note.controller.js



```
                return res.status(404).send({
                    message: "Note not found with id " + req.params.noteId
                });
            }
            return res.status(500).send({
                message: "Error updating note with id " + req.params.noteId
            });
        });
};

// Delete a note with the specified noteId in the request
exports.delete = (req, res) => {
    Note.findByIdAndRemove(req.params.noteId)
    .then(note => {
        if(!note) {
            return res.status(404).send({
                message: "Note not found with id " + req.params.noteId
            });
        }
        res.send({message: "Note deleted successfully!"});
    }).catch(err => {
        if(err.kind === 'ObjectId' || err.name === 'NotFound') {
            return res.status(404).send({
                message: "Note not found with id " + req.params.noteId
            });
```

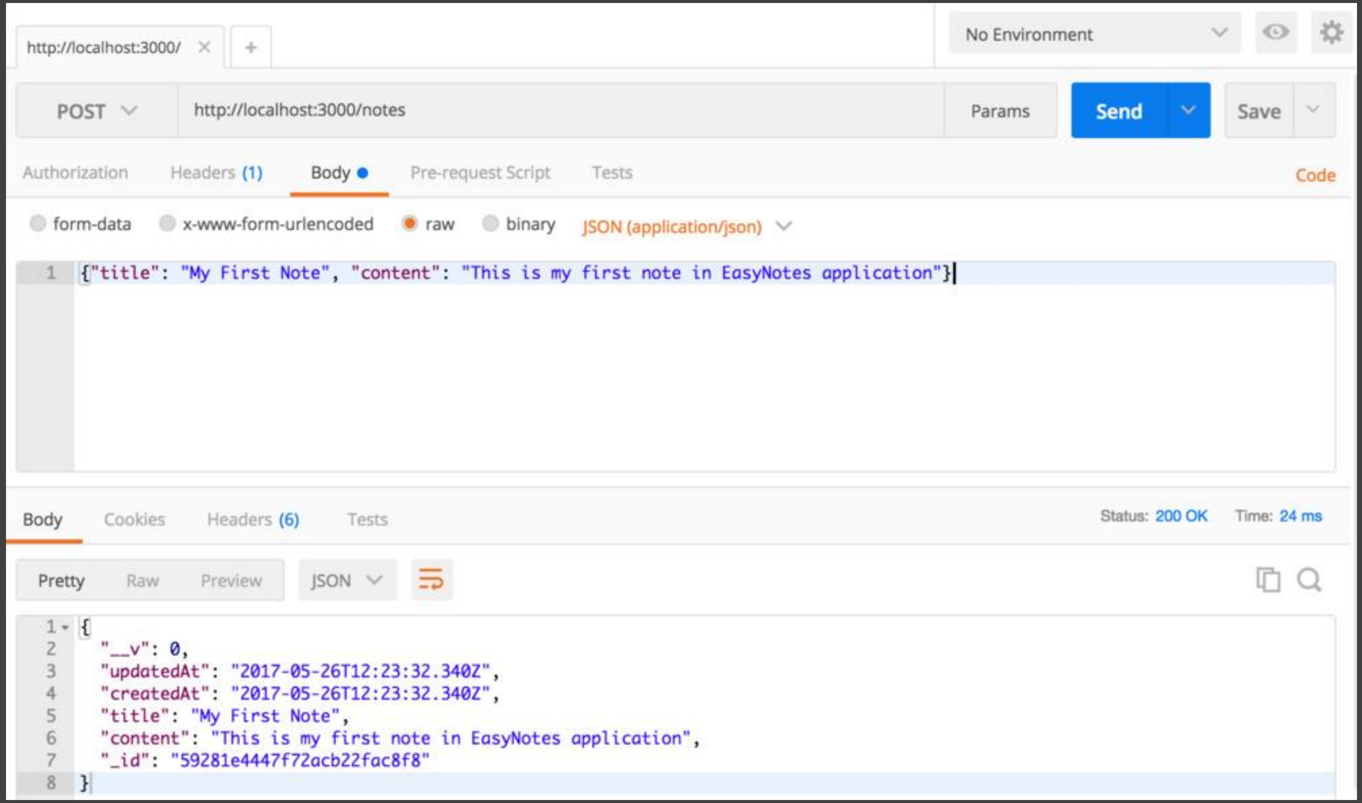
Controllers (6)

App.controllers -> note.controller.js



```
    }).catch(err => {
        if(err.kind === 'ObjectId' || err.name === 'NotFound') {
            return res.status(404).send({
                message: "Note not found with id " + req.params.noteId
            });
        }
        return res.status(500).send({
            message: "Could not delete note with id " + req.params.noteId
        });
    });
};
```

Creating a new Note
POST /notes API



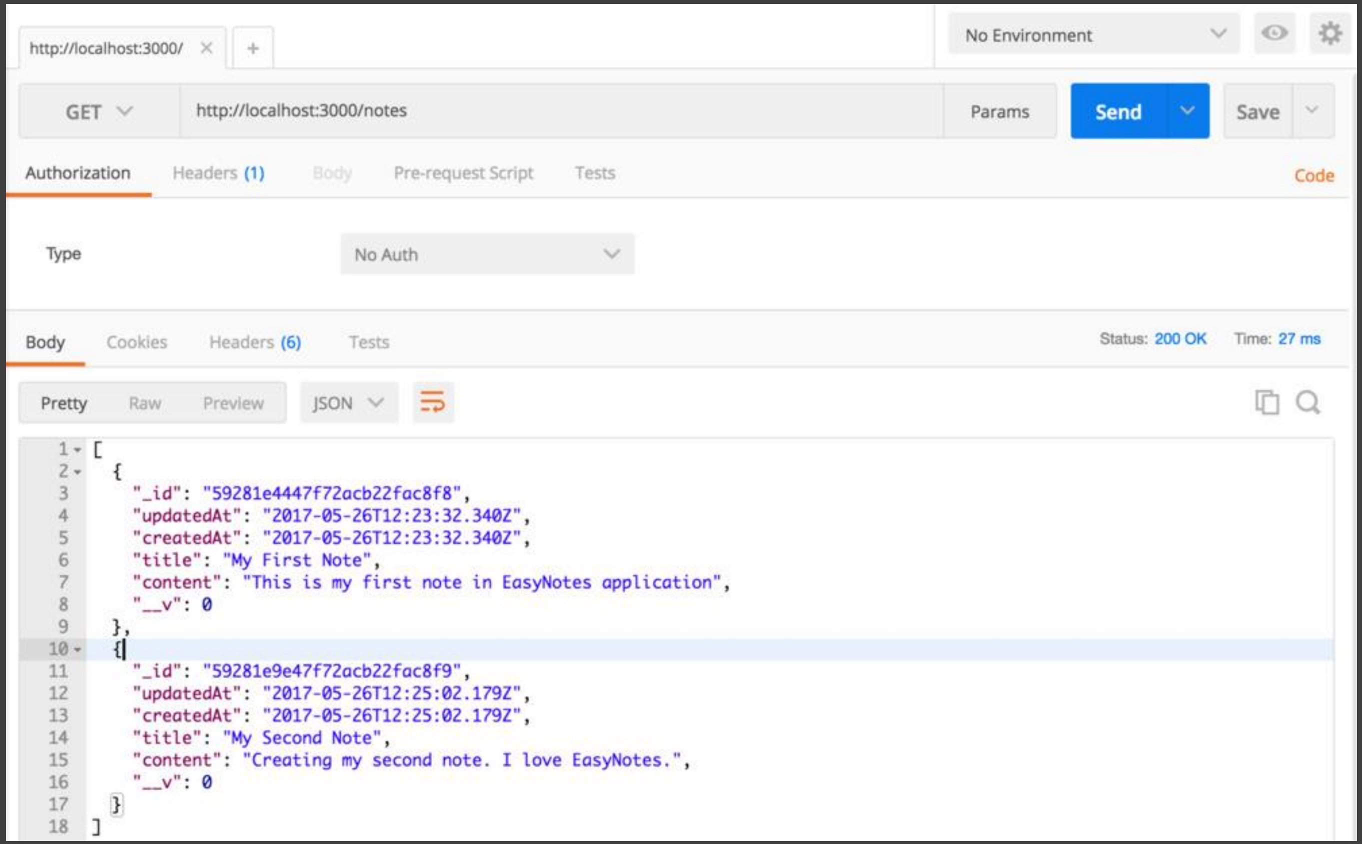
POST http://localhost:3000/notes

```
{"title": "My First Note", "content": "This is my first note in EasyNotes application"}
```

Status: 200 OK Time: 24 ms

```
{
  "__v": 0,
  "updatedAt": "2017-05-26T12:23:32.340Z",
  "createdAt": "2017-05-26T12:23:32.340Z",
  "title": "My First Note",
  "content": "This is my first note in EasyNotes application",
  "_id": "59281e4447f72acb22fac8f8"
}
```

Retrieving all Notes
GET /notes API



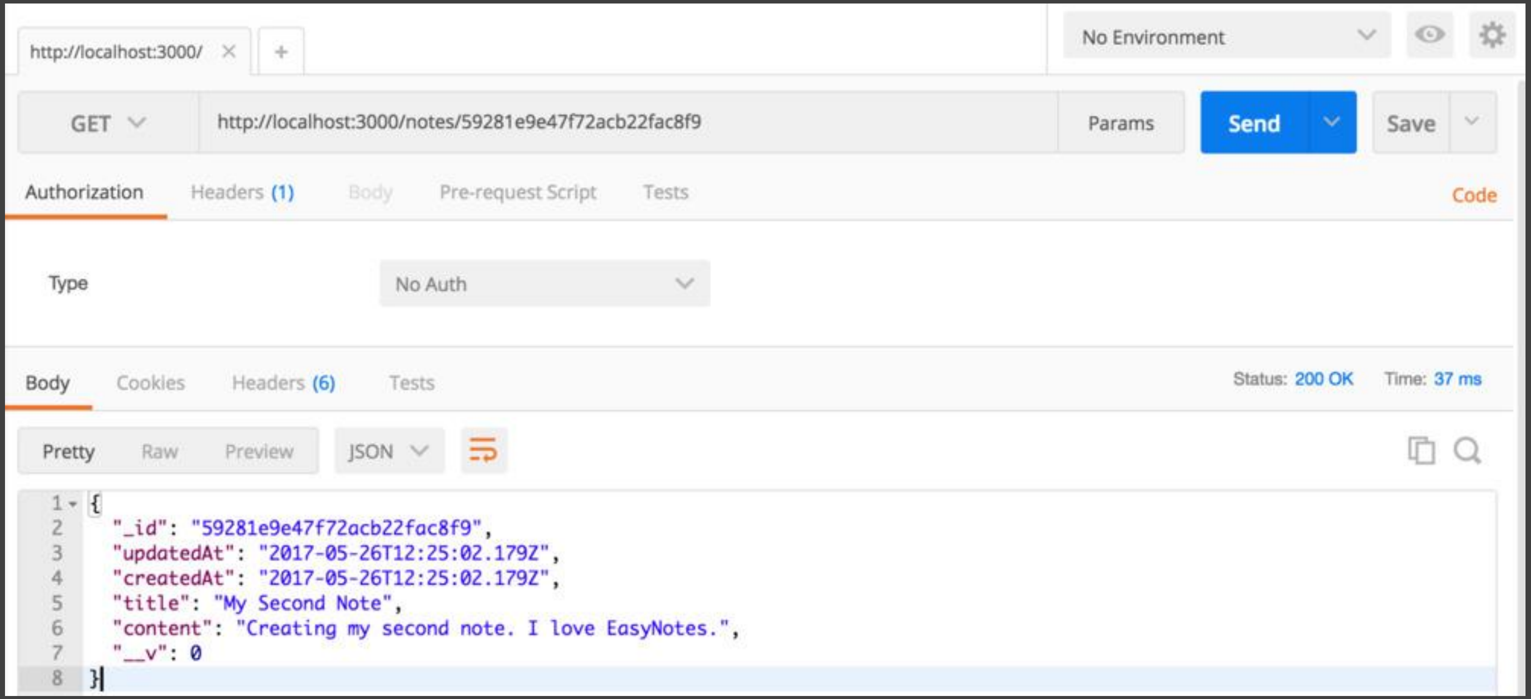
http://localhost:3000/notes

Type: No Auth

Status: 200 OK   Time: 27 ms

```
[
  {
    "_id": "59281e4447f72acb22fac8f8",
    "updatedAt": "2017-05-26T12:23:32.340Z",
    "createdAt": "2017-05-26T12:23:32.340Z",
    "title": "My First Note",
    "content": "This is my first note in EasyNotes application",
    "__v": 0
  },
  {
    "_id": "59281e9e47f72acb22fac8f9",
    "updatedAt": "2017-05-26T12:25:02.179Z",
    "createdAt": "2017-05-26T12:25:02.179Z",
    "title": "My Second Note",
    "content": "Creating my second note. I love EasyNotes.",
    "__v": 0
  }
]
```

Retrieving a single Note GET /notes/:noteId API



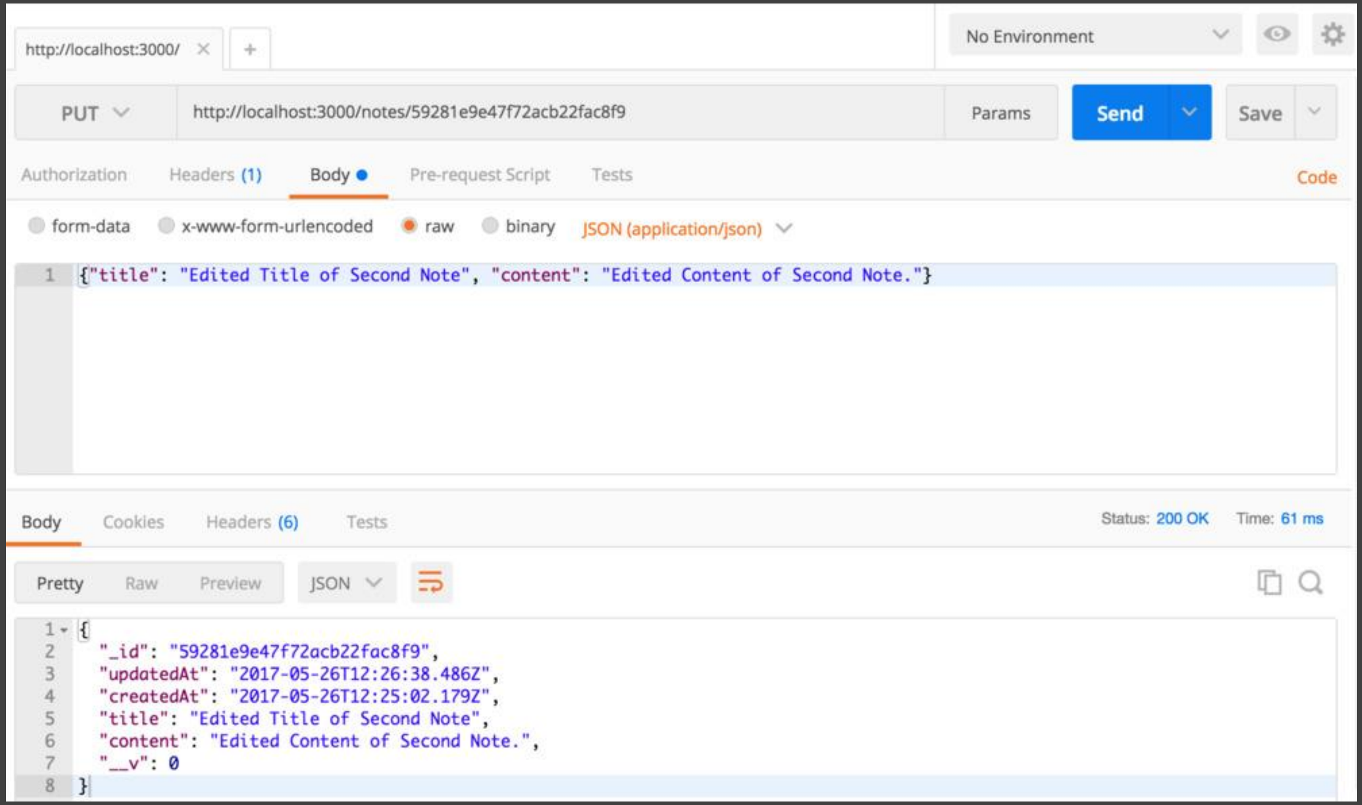
```
http://localhost:3000/notes/59281e9e47f72acb22fac8f9
```

Type: No Auth

Status: 200 OK Time: 37 ms

```
{
  "_id": "59281e9e47f72acb22fac8f9",
  "updatedAt": "2017-05-26T12:25:02.179Z",
  "createdAt": "2017-05-26T12:25:02.179Z",
  "title": "My Second Note",
  "content": "Creating my second note. I love EasyNotes.",
  "__v": 0
}
```

Updating a Note
PUT /notes/:noteId API



PUT http://localhost:3000/notes/59281e9e47f72acb22fac8f9

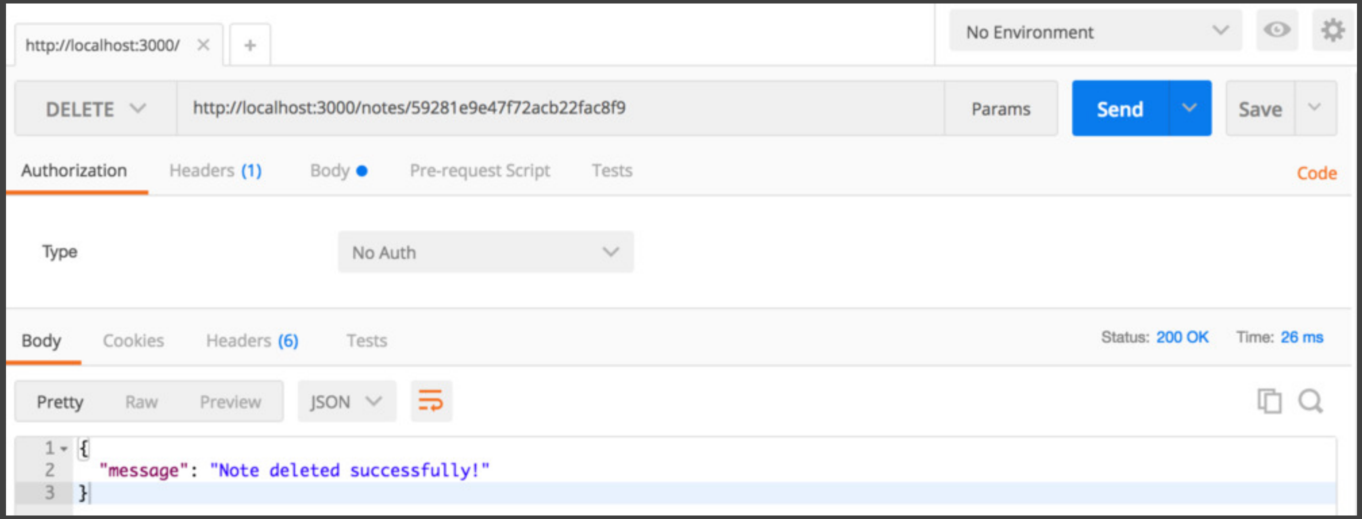
Body: raw, JSON (application/json)

```
{"title": "Edited Title of Second Note", "content": "Edited Content of Second Note."}
```

Status: 200 OK Time: 61 ms

```
{
  "_id": "59281e9e47f72acb22fac8f9",
  "updatedAt": "2017-05-26T12:26:38.486Z",
  "createdAt": "2017-05-26T12:25:02.179Z",
  "title": "Edited Title of Second Note",
  "content": "Edited Content of Second Note.",
  "__v": 0
}
```

Deleting a Note
DELETE /notes/:noteId API



DELETE http://localhost:3000/notes/59281e9e47f72acb22fac8f9

Type: No Auth

Status: 200 OK Time: 26 ms

JSON

```
{
  "message": "Note deleted successfully!"
}
```