```
const express = require("express");
const mongoose = require("mongoose");
const app = express();
app.use(express.json());

/* Connecting to MongoDB */
mongoose.connect("mongodb://localhost:27017/newdb", {
   useCreateIndex: true,
   useNewUrlParser: true,
   useUnifiedTopology: true
}).then(() => {
   console.log("Established Connection");
}).catch((e) => {
   console.error("Connection error", e);
});

/*Schema*/
const clientSchema = new mongoose.Schema({
   firstName: {
      type: String,
      required: [true, 'Input first name'],
      trim: true
   },
   lastName: {
      type: String,
      required: [true, 'Input last name'],
      trim: true
   },
   hobby: {
      type: String,
      required: [true, 'Input hobby'],
      trim: true
   }
});

/*Client model*/
const ClientModel = mongoose.model("Client", clientSchema);

/* Incoming request logger */
function L(req, res, next) {
   console.log(`Arrived request - Method: [${req.method}] Path: ${req.path}`);
   next();
}

/*validator*/
function validateInput(req, res, next) {
   const { firstName, lastName, hobby } = req.body;
   if (!firstName || !lastName || !hobby) {
      return res.status(400).json({
         error: 'Please provide required inputs'
```

```
        });
    }
    next();
}

app.use(L);

/*display all clients*/
app.get('/clients', async (req, res) => {
    try {
        const allClients = await ClientModel.find({});
        res.status(200).json(allClients);
    } catch (e) {
        res.status(500).send('Internal Server Error');
    }
});

/*get request*/
app.get('/clients/:id', async (req, res) => {
    try {
        const client = await ClientModel.findById(req.params.id);
        if (!client) {
            return res.status(404).json({ error: 'Invalid user' });
        }
        res.status(200).json(client);
    } catch (e) {
        res.status(500).send('Internal Server Error');
    }
});

app.post('/client', validateInput, async (req, res) => {
    try {
        const newClient = new ClientModel(req.body);
        const savedClient = await newClient.save();
        res.status(201).json(savedClient);
    } catch (e) {
        res.status(400).send('Bad Request');
    }
});

/*update*/
app.put('/client/:id', validateInput, async (req, res) => {
    try {
        const updatedClient = await ClientModel.findByIdAndUpdate(req.params.id, req.body, { new:
true, runValidators: true });
        if (!updatedClient) {
            return res.status(404).json({ error: 'Not found' });
        }
        res.status(200).json(updatedClient);
    } catch (e) {
        res.status(400).send('Bad Request');
    }
});

/*delete */
app.delete('/client/:id', async (req, res) => {
    try {
        const removedClient = await ClientModel.findByIdAndDelete(req.params.id);
        if (!removedClient) {
            return res.status(404).json({ error: "Doesn't exist" });
        }
        res.status(200).json(removedClient);
    } catch (e) {
        res.status(500).send('Internal Server Error');
    }
});

const port = process.env.PORT || 3000;
app.listen(port, () => {
    console.log(`Hoisted at http://localhost:${port}`);
});
```