WEEK 6

TYPE THE CODE

nodeapp > controllers > JS taskController.js > ...

```js
/*
    WEEK 6 - TYPE THE CODE
    controllers > taskController.js
*/

const Task = require("../models/taskModel");

exports.createTask = async(req, res) => {
    try {
        const {title, completed} = req.body;
        const task = new Task({title, completed});
        const savedTask = await task.save();
        res.status(201).json(savedTask);
    } catch(err) {
        res.status(500).json({message: err.message});
    }
};

exports.getAllTasks = async(req, res) => {
    try {
        res.status(200).json(await Task.find());
    } catch(err) {
        res.status(500).json({message: err.message});
    }
};

exports.getTaskById = async(req, res) => {
    try {
        const task = await Task.findById(req.params.id);
        if(!task) return res.status(404).json({message: "Task not found"});
        res.status(200).json(task);
    } catch(err) {
        res.status(500).json({message: err.message});
    }
}
```

MongoDB | JS index.js | JS taskRoutes.js M | JS taskController.js M | JS taskModel.js M

nodeapp > routers > JS taskRoutes.js > ...

```js
/*
    WEEK 6 - TYPE THE CODE
    routers > taskRoutes.js
*/

const express = require('express');
const {createTask, getAllTasks, getTaskById} = require('../controllers/taskController');

const router = express.Router();
router.post("/", createTask);
router.get("/", getAllTasks);
router.get("/:id", getTaskById);

module.exports = router;
```

MongoDB | index.js | taskRoutes.js | taskController.js M

nodeapp > models > taskModel.js > ...

```
/*
    WEEK 6 - TYPE THE CODE
    models > taskModel.js
*/

const mongoose = require('mongoose');

const taskSchema = new mongoose.Schema({
    title: {
        type: String,
        required: true
    },
    completed: {
        type: Boolean,
        default: false
    }
});

module.exports = mongoose.model("Task", taskSchema);
```

MongoDB | index.js M | taskRoutes.js M | taskController.js M | taskModel.js M

nodeapp > index.js > ...

```
/*
    WEEK 6 - TYPE THE CODE
    index.js
*/

const express = require('express');
const mongoose = require('mongoose')
const taskRoutes = require('./routers/taskRoutes');

const app = express();
app.use(express.json());
app.use("/tasks",taskRoutes);

mongoose.connect("mongodb://127.0.0.1:27017/taskdb")
    .then(() => app.listen(8080, () => console.log("Server running on 8080")))
    .catch(err => console.error(err))
```

FIX THE CODE

nodeapp > JS index.js > ...

```
// WEEK 6 - Fix the Code

const express = require('express');
const mongoose = require('mongoose');
const itemRoutes = require('./routers/itemRouter');

const app = express();

app.use(express.json());

mongoose.connect('mongodb://127.0.0.1:27017/itemDB', {
  useNewUrlParser: true,
  useUnifiedTopology: true
})
  .then(() => console.log('Connected to MongoDB'))
  .catch(err => console.log(err));

app.use('/items', itemRoutes);

const port = 8080;
app.listen(port, () => {
  console.log('Server is running on port ' + port);
});
```

nodeapp > routers > JS itemRouter.js > ...

```
// WEEK 6 - Fix the Code

const { createItem, getAllItems } = require('../controllers/itemController');

const router = express.Router();

router.post('/', createItem);

router.get('/', getAllItems);

module.exports = router;
```

nodeapp > controllers > JS itemController.js > ...

```
// WEEK 6 - Fix the Code

const Item = require('../models/itemModel');
const createItem = async (req, res) => {
  const { name, quantity } = req.body;

  if (!name || quantity == null) {
    return res.status(400).json({ message: 'Name and quantity is required' });
  }

  try {
    const newItem = new Item({
      name, quantity
    });

    const savedItem = await newItem.save();
    res.status(201).json(savedItem);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
};

const getAllItems = async (req, res) => {
  try {
    const items = await Item.find();
    res.status(200).json(items);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
};

module.exports = {
  createItem,
  getAllItems
};

```

nodeapp > models > JS itemModel.js > ...

```
// WEEK 6 - Fix the Code

const mongoose = require('mongoose');

const itemSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  quantity: {
    type: Number,
    required: true
  }
});

const Item = mongoose.model('Item', itemSchema);

module.exports = Item;
```

PRACTICE AT HOME 1

nodeapp > JS index.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 1
    index.js
*/

const express = require('express');
const mongoose = require('mongoose');
const productRoutes = require('./routers/productRoutes');
const errorHandler = require('./middlewares/errorHandler');

const app = express();
app.use(express.json());

mongoose.connect("mongodb://127.0.0.1:27017/productsDB", {
    useNewUrlParser: true,
    useUnifiedTopology: true
})
.then(() => console.log('MongoDB connected successfully'))
.catch((err)=> console.error(err));

app.use('./products', productRoutes);
app.use(errorHandler);

app.listen(8080, ()=>{
    console.log('Server running on port 8080');
});
```

nodeapp > routers > JS productRoutes.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 1
    routers > productRoutes.js
*/

const express = require('express');
const router = express.Router();
const {createProduct, getAllProducts} = require('../controllers/productController');

router.post('/', createProduct);
router.get('/', getAllProducts);

module.exports = router;
```

nodeapp > models > JS productModel.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 1
    models > productModel.js
*/

const mongoose = require('mongoose');
const productSchema = new mongoose.Schema({
    name: {
        type: String,
        required: [true, 'Product name is required']
    },
    price: {
        type: Number,
        required: [true, 'Product price is required']
    },
    category: {
        type: String,
        required: [true, 'Product category is required']
    },
    inStock: {
        type: Boolean,
        default: true
    }
}, {timestamps: true});

module.exports = mongoose.model('Product', productSchema);
```

nodeapp > middlewares > JS errorHandler.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 1
    middlewares > errorHandler.js
*/

const errorHandler = (err, req, res, next) => {
    console.error(err.stack);
    const statusCode = 500;
    res.status(statusCode).json({
        error: {
            message: err.message,
            stack: process.env.NODE_ENV === 'production' ? null : err.stack
        }
    });
};

module.exports = errorHandler;
```

nodeapp > controllers > JS productController.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 1
    controllers > productController.js
*/

const Product = require('../models/productModel');

const createProduct = async(req, res, next) => {
    try {
        const {name, price, category, inStock} = req.body;
        if(!name || !price || !category) {
            throw new Error('Database Error');
        }
        const product = new Product({name, price, category, inStock});
        const savedProduct = await product.save();
        res.status(201).json(savedProduct);
    } catch(err) {
        next(err);
    }
};

const getAllProducts = async(req, res, next) => {
    try {
        const products = await Product.find();
        res.status(200).json(products);
    } catch(err) {
        next(err);
    }
};

module.exports = {createProduct, getAllProducts};
```

PRACTICE AT HOME 2

JS movieController.js M | JS movieRoutes.js M | JS movieModel.js M | JS index.js M

nodeapp > routers > JS movieRoutes.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 2
    routers > movieRoutes.js
*/

const ex = require('express');
const r = ex.Router();
const {cm, gm, gmi, um, dm} = require('../controllers/movieController');

r.post('/movies', cm);
r.get('/movies',gm);
r.get('/movies/:id', gmi);
r.put('/movies/:id', um);
r.delete('/movies/:id',dm);

module.exports = r;
```

JS movieController.js M | JS movieRoutes.js M | JS movieModel.js M | JS index.js M ×

nodeapp > JS index.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 2
    index.js
*/

const ex = require('express');
const mg = require('mongoose');
const mr = require('./routers/movieRoutes');

const app = ex();
app.use(ex.json());

mg.connect('mongodb://127.0.0.1:27017/movieDB')
    .catch(e => console.log(e));

app.use('/movies', mr);
app.listen(8080);
```

nodeapp > models > JS movieModel.js > ...

```
/*
    WEEK 6 - PRACTICE AT HOME 2
    models > movieModels.js
*/

const mg = require('mongoose');
const ms = new mg.Schema({
    title: {
        type: String,
        required: true
    },
    director: {
        type: String,
        required: true
    },
    releaseYear: {
        type: Number,
        required: true
    },
    genre: {
        type: String,
        required: true
    }
});

const m = mg.model('Movie', ms);
module.exports = m;
```

JS movieController.js M × | JS movieRoutes.js M | JS movieModel.js M | JS index.js M

nodeapp > controllers > JS movieController.js > getMovieById

```
/*
    WEEK 6 - PRACTICE AT HOME 2
    controller > movieController.js
*/

const Movie = require('../models/movieModel');
const createMovie = async(req, res) => {
    const {title, director, releaseYear, genre} = req.body;
    try {
        const newMovie = new Movie({
            title, director, releaseYear, genre
        });

        const savedMovie = await newMovie.save();
        res.status(201).json(savedMovie);
    } catch (err){ res.status(500).json({message: err.message}); }
};

const getAllMovies = async(req, res) => {
    try {
        const movies = await Movie.find();
        res.status(200).json(movies);
    } catch(err) { res.status(500).json({message: err.message});}
};

const getMovieById = async(req, res) => {
    try {
        const movie = await Movie.findById(req.params.id);
        if(!movie) { return res.status(404).json({message: "Movie not found"}); }
        res.status(200).json(movie);
    } catch(err) { res.status(500).json({message: err.message}); }
};

const updateMovie = async(req, res) => {
    try {
        const movie = await Movie.findByIdAndUpdate(req.params.id, req.body, {new: true, runValidators: true});
        if(!movie) {
            return res.status(404).json({message: "Movie not found"});
        }

        res.status(200).json(movie);
    } catch(err) {
        res.status(500).json({message: err.message});
    }
};

const deleteMovie = async(req, res) => {
    try {
        const movie = await Movie.findByIdAndDelete(req.params.id);
        if(!movie) {
            return res.status(404).json({message: "Movie not found"});
        }

        res.status(200).json({message: "Movie deleted successfully"});
    } catch(err) {
        res.status(500).json({message: err.message});
    }
};

module.exports = {createMovie, getAllMovies, getMovieById, updateMovie, deleteMovie};
```

CHALLENGE YOURSELF

nodeapp > controllers > JS customerController.js > ...

```
/*
    WEEK 6 - CHALLENGE YOURSELF
    controllers > customerController.js
*/

const Cust = require('../models/customerModel');
const createCustomer = async(req, res, next) => {
    try{
        const {name, email, phone, isActive} = req.body;
        if(!name || !email || !phone) return res.status(404).json({message: "Name, email, and phone are required"});
        const c = new Cust({name, email, phone, isActive});
        const s = await c.save();
        res.status(201).json(s);
    } catch(err) { next(err); }
};

const getAllCustomers = async(req, res, next) => {
    try { res.status(200).json(await Cust.find());} catch(err) { next(err);}
};

const updateCustomer = async(req, res, next) => {
    try {
        const u = await Cust.findByIdAndUpdate(req.params.id, req.body, {new: true});
        if(!req.body.name || !req.body.email || !req.body.phone)
            return res.status(404).json({message: "Name, email, and phone are required to update the customer"});
        if(!u) return res.status(404).json({message: "Customer not found"});
        res.status(200).json(u);
    } catch(err) { next(err); }
};

module.exports ={createCustomer, getAllCustomers, updateCustomer};
```

nodeapp > middlewares > JS errorHandler.js > ...

```
/*
    WEEK 6 - CHALLENGE YOURSELF
    middlewares > errorHandler.js
*/

const eh = (err, req, res, next) => {
    console.error(err.message);
    res.status(500).json({message: "Server Error"});
};

module.exports = eh;
```

nodeapp > models > JS customerModel.js > ...

```
/*
    WEEK 6 - CHALLENGE YOURSELF
    models > customerModel.js
*/

const mg = require("mongoose");
const cs = new mg.Schema({
    name: {
        type: String,
        required: true
    },
    email: {
        type: String,
        required: true
    },
    phone: {
        type: String,
        required: true
    },
    isActive: {
        type: Boolean,
        default: true
    }
});

module.exports = mg.model("Customer", cs);

```

nodeapp > routers > JS customerRoutes.js > ...

```
/*
    WEEK 6 - CHALLENGE YOURSELF
    routers > customerRoutes.js
*/

const ex = require('express')
const {cc, gc, uc} = require("../controllers/customerController");
const r = ex.Router()

r.post('/customers', cc);
r.get('/customers', gc);
r.put('/customer/:id', uc);

module.exports = r;

```

nodeapp > JS index.js > ...

```
/*
    WEEK 6 - CHALLENGE YOURSELF
    index.js
*/

const ex = require('express');
const mg = require('mongoose');
const r = require('./routers/customerRoutes');
const eh = require('./middlewares/errorHandler');

const app = ex();
app.use(ex.json());

mg.connect("mongodb://127.0.0.1:27017/customerDB")
    .then(() => console.log("DB connected"))
    .catch(err => console.error(err));

app.use('/api', r);
app.use(eh);

app.listen(8080, () => console.log("Server running on 8080"));
```