
💼 Folder Structure

```
/billsplitter
│── /models
│   ├── Bill.js
│── /controllers
│   ├── billController.js
│── /routes
│   ├── billRoutes.js
│── server.js
```

📜 Mongoose Model (Bill.js)

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

const BillSchema = new mongoose.Schema({
  billName: String,
  totalAmount: Number,
  date: { type: Date, default: Date.now },
  participants: [
    {
      name: String,
      amountOwed: Number,
      amountPaid: { type: Number, default: 0 },
    }
  ]
});

const Bill = mongoose.model("Bill", BillSchema);
module.exports = Bill;
```

📂 Controllers (billController.js)

```
const Bill = require("../models/Bill");

// ✅ Create a Bill
exports.createBill = async (req, res) => {
  let { billName, totalAmount, participants } = req.body;

  if (participants.every((p) => !p.amountOwed)) {
    const equalShare = totalAmount / participants.length;
    participants = participants.map((p) => ({
      ...p,
      amountOwed: equalShare,
    }));
  }

  const bill = new Bill({ billName, totalAmount, participants });
  await bill.save();
  res.status(201).json(bill);
};

// ✅ Get All Bills
exports.getBills = async (req, res) => {
  const bills = await Bill.find();
  res.json(bills);
};

// ✅ Get Single Bill
exports.getBillById = async (req, res) => {
  const bill = await Bill.findById(req.params.id);
  if (!bill) return res.status(404).json({ error: "Bill not found" });
  res.json(bill);
};

// ✅ Update a Bill
exports.updateBill = async (req, res) => {
  const bill = await Bill.findByIdAndUpdate(req.params.id, req.body, { new:
true });
  if (!bill) return res.status(404).json({ error: "Bill not found" });
  res.json(bill);
};

// ✅ Delete a Bill
exports.deleteBill = async (req, res) => {
  const bill = await Bill.findByIdAndDelete(req.params.id);
  if (!bill) return res.status(404).json({ error: "Bill not found" });
  res.json({ message: "Bill deleted successfully" });
};

// ✅ Get Balances
exports.getBalances = async (req, res) => {
  const bill = await Bill.findById(req.params.id);
  if (!bill) return res.status(404).json({ error: "Bill not found" });

  const balances = bill.participants.map((p) => ({
    name: p.name,
    amountOwed: p.amountOwed,
    amountPaid: p.amountPaid,
    balance: p.amountOwed - p.amountPaid,
  }));

  res.json(balances);
};

// ✅ Get People Who Have Paid in Full
exports.getPaidUsers = async (req, res) => {
  const bill = await Bill.findById(req.params.id);
  if (!bill) return res.status(404).json({ error: "Bill not found" });

  const paidParticipants = bill.participants
    .filter((p) => p.amountPaid >= p.amountOwed)
    .map((p) => ({
      name: p.name,
      status: p.amountPaid > p.amountOwed ? "Overpaid" : "Paid in Full",
    }));

  res.json(paidParticipants);
};

// ✅ Get Total Amount Paid
exports.getTotalPaid = async (req, res) => {
  const bill = await Bill.findById(req.params.id);
  if (!bill) return res.status(404).json({ error: "Bill not found" });

  const totalPaid = bill.participants.reduce((sum, p) => sum + p.amountPaid,
0);
  res.json({ totalPaid });
};

// ✅ Get Top Payer
exports.getTopPayer = async (req, res) => {
  const bill = await Bill.findById(req.params.id);
  if (!bill) return res.status(404).json({ error: "Bill not found" });

  const topPayer = bill.participants.reduce((top, p) =>
    p.amountPaid > top.amountPaid ? p : top
  );

  res.json(topPayer);
};
```

📂 Routes (billRoutes.js)

```
const express = require("express");
const router = express.Router();
const billController = require("../controllers/billController");

router.post("/", billController.createBill);
router.get("/", billController.getBills);
router.get("/:id", billController.getBillById);
router.put("/:id", billController.updateBill);
router.delete("/:id", billController.deleteBill);

router.get("/:id/balances", billController.getBalances);
router.get("/:id/paid", billController.getPaidUsers);
router.get("/:id/total-paid", billController.getTotalPaid);
router.get("/:id/top-payer", billController.getTopPayer);

module.exports = router;
```

🌍 Server Setup (server.js)

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

mongoose.connect("mongodb://127.0.0.1:27017/billsplitter", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

app.use("/bills", billRoutes);

app.listen(3000, () => console.log("Server running on port 3000"));
```

🔥 API Endpoints

Method	Endpoint	Description
POST	`/bills`	Create a new bill
GET	`/bills`	Get all bills
GET	`/bills/:id`	Get a specific bill
PUT	`/bills/:id`	Update a bill
DELETE	`/bills/:id`	Delete a bill
GET	`/bills/:id/balances`	Get how much each participant owes
GET	`/bills/:id/paid`	Get users who have paid in full
GET	`/bills/:id/total-paid`	Get the total amount paid
GET	`/bills/:id/top-payer`	Get the biggest payer

📌 Example Usage

1️⃣ Create a Bill

```
{
  "billName": "Netflix Subscription",
  "totalAmount": 6000,
  "participants": [
    { "name": "Alice", "amountPaid": 3000 },
    { "name": "Bob", "amountPaid": 1000 },
    { "name": "Charlie", "amountPaid": 2000 }
  ]
}
```

2️⃣ Get Balances

```
[
  { "name": "Alice", "amountOwed": 2000, "amountPaid": 3000, "balance": -1000
},
  { "name": "Bob", "amountOwed": 2000, "amountPaid": 1000, "balance": 1000 },
  { "name": "Charlie", "amountOwed": 2000, "amountPaid": 2000, "balance": 0 }
]
```

🎯 Features Covered

✅ **MVC Structure (Model, Controller, Routes, Server)**
✅ **CRUD Operations**
✅ **Money-Based Calculations (Owed, Paid, Balances, Overpayments)**
✅ **Proper Express.js Controller Separation**