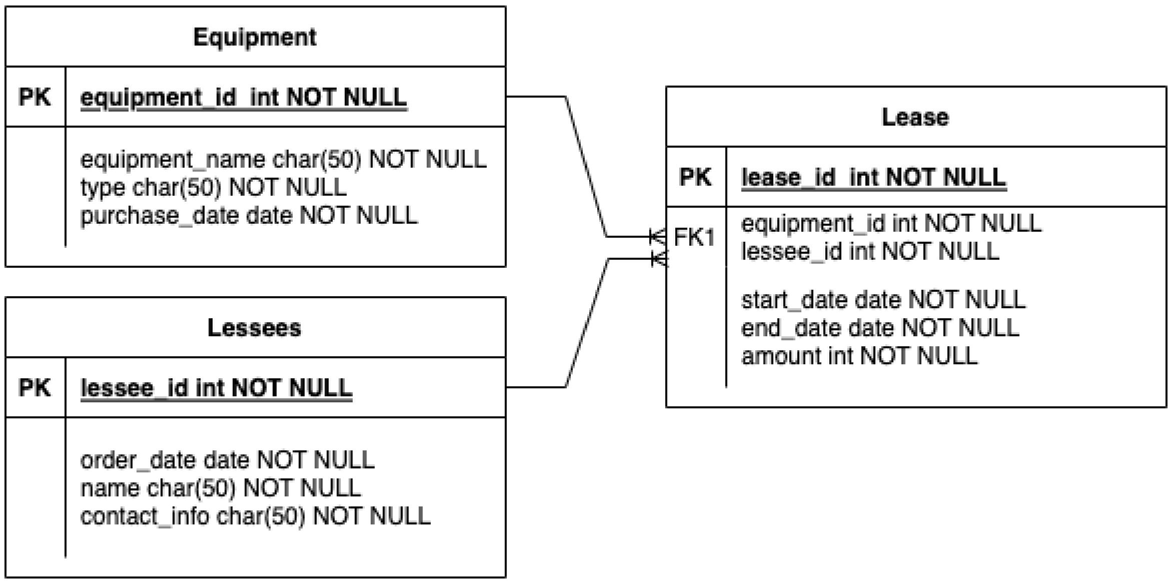
Case assignment - Leasi

Database Design

Relational Database

RDBMS Schema



Relationships:

- **Equipment to Lease**: One-to-Many (An equipment can be part of multiple leases, but a lease is associated with only one equipment).
- **Lessees to Lease**: One-to-Many (A lessee can have multiple leases, but a lease is associated with only one lessee).

Document Database

I would use MongoDB (NoSQL) for Document Database.

NoSQL Schema

```
{
  "_id": ObjectId("..."),
  "name": "name",
  "type": "type",
  "purchase_date": ISODate("2023-01-15"),
}
```

```
{
  "_id": ObjectId("..."),
  "name": "name",
  "contact_info": "contact@example.com",
}
```

```
{
  "_id": ObjectId("..."),
  "equipment": {
    "equipment_id": ObjectId("..."),
    "name": "name"
  },
  "lessee": {
    "lessee_id": ObjectId("..."),
    "name": "name"
  },
  "start_date": ISODate("2023-02-01"),
  "end_date": ISODate("2023-03-01"),
  "amount": 500,
}
```

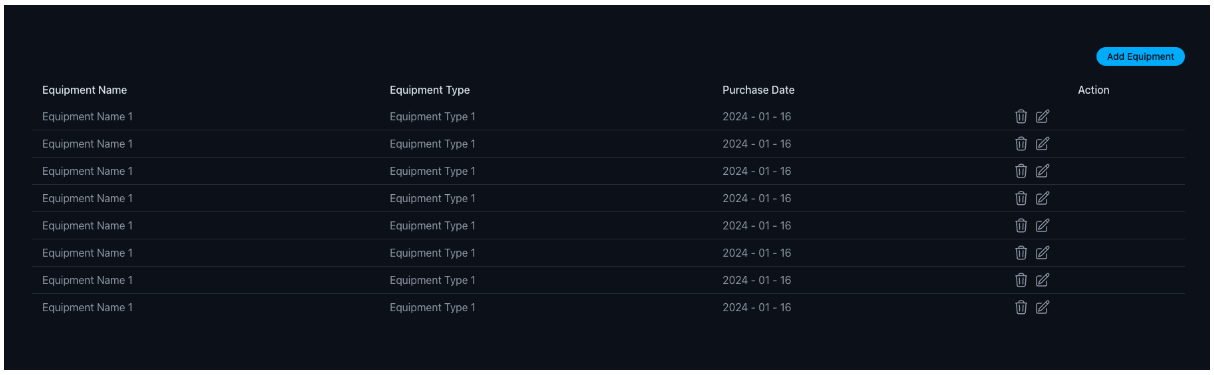
APIs Design

```
const express = require('express');
const bodyParser = require('body-parser');
const app = express();
const PORT = 3000;

app.use(bodyParser.json());
```

```
// Get all equipment
app.get('/api/equipment', (req, res) => {
    res.json(equipmentList);
});

// Get equipment by ID
app.get('/api/equipment/:id', (req, res) => {
    const equipmentId = req.params.id;
    const equipment = equipmentList.find(item => item.id ===

    if (!equipment) {
        return res.status(404).json({ message: 'Equipment not
    }

    res.json(equipment);
});

// Create new equipment
app.post('/api/equipment', (req, res) => {
    const newEquipment = req.body;
    equipmentList.push(newEquipment);
    res.json(newEquipment);
});

// Update equipment
app.put('/api/equipment/:id', (req, res) => {
    const equipmentId = req.params.id;
    const updatedEquipment = req.body;

    const index = equipmentList.findIndex(item => item.id ===

    if (index === -1) {
        return res.status(404).json({ message: 'Equipment not
    }

    equipmentList[index] = { ...equipmentList[index], ...upda

    res.json(equipmentList[index]);
```

```
});

// Delete equipment
app.delete('/api/equipment/:id', (req, res) => {
    const equipmentId = req.params.id;

    equipmentList = equipmentList.filter(item => item.id !==

        if (index === -1) {
        return res.status(404).json({ message: 'Equipment not
    }

    res.json({ message: 'Equipment deleted successfully' });
});

app.listen(PORT, () => {
    console.log(`Server is running on port ${PORT}`);
});
```

User Interface Design



Add Equipment

Equipment Name	Equipment Type	Purchase Date	Action
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎
Equipment Name 1	Equipment Type 1	2024 - 01 - 16	🗑 ✎