**1. Frontend**

* **Tech Stack**: React with Vite for faster builds.
* **Features**:
  + Search bar to input the license number.
  + Display results in a table.
  + Basic styling with CSS or a library like TailwindCSS/Bootstrap for responsiveness.

**2. Backend**

* **Tech Stack**: Node.js with Express for routing.
* **Features**:
  + API endpoints to:
    - Search license by number.
    - Perform CRUD operations for the admin dashboard.
  + Database (e.g., MongoDB or PostgreSQL) to store license data.

**3. Admin Dashboard**

* **Features**:
  + Add, edit, delete, and view license records.
  + Authentication for admin users.

**Next Steps**

* **Set Up the Project**: Do you want guidance on setting up the React and Node.js environment?
* **Database**: Which database would you like to use (MongoDB, MySQL, etc.)?

Let’s break it down further if needed! 😊

…………………………………………………………………………………………………………………………………………………………………

Perfect! MongoDB is a great choice for this project. Here's an updated breakdown with MongoDB:

**1. Backend (Node.js + Express + MongoDB)**

* **Dependencies**:
  + express for routing.
  + mongoose to interact with MongoDB.
  + dotenv for environment variables.
  + cors to handle cross-origin requests.
* **Key API Endpoints**:
  + POST /api/licenses to add a new license.
  + GET /api/licenses/:licenseNo to retrieve a license by number.
  + PUT /api/licenses/:id to update license details.
  + DELETE /api/licenses/:id to delete a license.

**2. Admin Dashboard**

* **Features**:
  + **CRUD Operations**:
    - Form for adding/editing licenses.
    - Table showing all license records with options to edit/delete.
  + **Admin Authentication**:
    - Use jsonwebtoken (JWT) for secure access.

**3. Frontend (React + Vite)**

* **Components**:
  + **SearchBar**: For license search input.
  + **LicenseTable**: Displays search results in a table.
  + **AdminDashboard**: Manages CRUD operations.

**Next Steps**

1. **Set up the backend**: Start with Express and connect it to MongoDB.
2. **Build the frontend**: Use React for the interface.
3. **Connect frontend to backend**: Use axios or fetch for API calls.

Would you like help with setting up the project structure, or should we dive into coding specific parts? 🚀

### ****Step 1: Set Up the Backend****

#### 1.1. ****Initialize the Project****

Run the following commands to create the backend folder:

mkdir driving-license-backend

cd driving-license-backend

npm init –y

#### 1.2. ****Install Dependencies****

Install the required Node.js packages:

npm install express mongoose cors dotenv body-parser

npm install --save-dev nodemon

#### 1.3. ****Project Structure****

Organize your files like this:

driving-license-backend/

├── .env

├── server.js

├── models/

│ └── License.js

├── routes/

│ └── licenseRoutes.js

└── config/

└── db.js

1.4. **MongoDB Connection (config/db.js)**

const mongoose = require('mongoose');

const connectDB = async () => {

try {

await mongoose.connect(process.env.MONGO\_URI, {

useNewUrlParser: true,

useUnifiedTopology: true,

});

console.log('MongoDB connected successfully');

} catch (err) {

console.error('MongoDB connection failed:', err.message);

process.exit(1);

}

};

module.exports = connectDB;

1.5. **License Model (models/License.js)**

const mongoose = require('mongoose');

const LicenseSchema = new mongoose.Schema({

licenseNumber: { type: String, required: true, unique: true },

name: { type: String, required: true },

dateOfBirth: { type: String, required: true },

issueDate: { type: String, required: true },

expiryDate: { type: String, required: true },

status: { type: String, required: true }, // e.g., "Valid" or "Expired"

});

module.exports = mongoose.model('License', LicenseSchema);

1.6. **API Routes (routes/licenseRoutes.js)**

const express = require('express');

const License = require('../models/License');

const router = express.Router();

// Add a new license

router.post('/', async (req, res) => {

try {

const license = new License(req.body);

await license.save();

res.status(201).json(license);

} catch (err) {

res.status(400).json({ error: err.message });

}

});

// Get a license by license number

router.get('/:licenseNumber', async (req, res) => {

try {

const license = await License.findOne({ licenseNumber: req.params.licenseNumber });

if (!license) return res.status(404).json({ error: 'License not found' });

res.json(license);

} catch (err) {

res.status(500).json({ error: err.message });

}

});

// Update a license

router.put('/:id', async (req, res) => {

try {

const updatedLicense = await License.findByIdAndUpdate(req.params.id, req.body, { new: true });

if (!updatedLicense) return res.status(404).json({ error: 'License not found' });

res.json(updatedLicense);

} catch (err) {

res.status(400).json({ error: err.message });

}

});

// Delete a license

router.delete('/:id', async (req, res) => {

try {

const deletedLicense = await License.findByIdAndDelete(req.params.id);

if (!deletedLicense) return res.status(404).json({ error: 'License not found' });

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

} catch (err) {

res.status(500).json({ error: err.message });

}

});

module.exports = router;

1.7. **Server Setup (server.js)**

const express = require('express');

const cors = require('cors');

const bodyParser = require('body-parser');

const dotenv = require('dotenv');

const connectDB = require('./config/db');

const licenseRoutes = require('./routes/licenseRoutes');

// Load environment variables

dotenv.config();

// Connect to MongoDB

connectDB();

const app = express();

// Middleware

app.use(cors());

app.use(bodyParser.json());

// Routes

app.use('/api/licenses', licenseRoutes);

// Start the server

const PORT = process.env.PORT || 5000;

app.listen(PORT, () => {

console.log(`Server running on http://localhost:${PORT}`);

});

1.8. **Environment File (.env)**

MONGO\_URI=mongodb+srv://<username>:<password>@cluster.mongodb.net/driving-licenses

PORT=5000

#### 1.9. ****Start the Server****

Add a script in package.json for development:

"scripts": {

"start": "node server.js",

"dev": "nodemon server.js"

}

**Run the server:**

npm run dev