Step 2: models/User.js

```js
import mongoose from "mongoose";

// Schema define karte hain
const UserSchema = new mongoose.Schema({
  name: { type: String, required: true },
  email: { type: String, required: true, unique: true },
  role: { type: String, default: "User" }
});

// Model banate hain
const User = mongoose.model("User", UserSchema);

export default User;
```

Step 1: Project Setup

```bash
mkdir crud-es6
cd crud-es6
npm init -y
npm install express mongoose dotenv
```

`package.json` me ek line add karna:

```json
"type": "module"
```

`.env` file banao:

```env
MONGO_URI=your_mongodb_atlas_uri
PORT=5000
```

Step 3: config/db.js

js

Copy code

```
import mongoose from "mongoose";

const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGO_URI, {
      useNewUrlParser: true,
      useUnifiedTopology: true
    });
    console.log("MongoDB Connected 🚀");
  } catch (err) {
    console.error("Connection Failed ❌", err);
    process.exit(1);
  }
};

export default connectDB;
```

Step 4: server.js

js

Copy code

```
import express from "express";
import dotenv from "dotenv";
import connectDB from "./config/db.js";
import User from "./models/User.js";

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

// MongoDB se connect karo
connectDB();
```

```
// Create (C)
app.post("/users", async (req, res) => {
  try {
    const user = await User.create(req.body);
    res.status(201).json(user);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```

```
// Read All (R)
app.get("/users", async (req, res) => {
  try {
    const users = await User.find();
    res.json(users);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// Read One (R by ID)
app.get("/users/:id", async (req, res) => {
  try {
    const user = await User.findById(req.params.id);
    res.json(user);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```

```
// Update (U)
app.put("/users/:id", async (req, res) => {
  try {
    const user = await User.findByIdAndUpdate(
      req.params.id,
      req.body,
      { new: true }
    );
    res.json(user);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// Delete (D)
app.delete("/users/:id", async (req, res) => {
  try {
    await User.findByIdAndDelete(req.params.id);
    res.send("User delete ho gaya ✅");
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```

```
// Server Run
const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(`Server chal raha hai: http://localhost:${PORT} 🔥`);
});
```

📂 Step 5: Postman se Test karo

1. **POST** `/users`

```json
{
  "name": "Rahul Sharma",
  "email": "rahul@example.com",
  "role": "Frontend Developer"
}
```

2. **GET** `/users` → sabhi users.
3. **GET** `/users/:id` → ek specific user.
4. **PUT** `/users/:id` → update user.
5. **DELETE** `/users/:id` → delete user.

🎯 Desi Samajh

- **ES6 import/export** = jaise tum ek almirah me samaan ko achhe se arrange karke label laga do, sab clean aur modern lagega.
- CRUD = Register book jisme doston ka record add, dekh, update, aur delete kar sakte ho.