**# Job Recruitment Platform Backend (Node.js & MongoDB)**

**Project Overview**

This is a **Job Recruitment Platform backend** built using **Node.js, Express, and MongoDB**. It allows job seekers to find jobs and apply, while employers can post job listings and manage applications.

---

**Folder Structure**

```
job-platform-backend/
|— node_modules/           # Installed dependencies
|— config/
|   └— db.js               # MongoDB connection setup
|— models/
|   ├— User.js             # User model (Job Seeker & Employer)
|   ├— Job.js              # Job model (Job Listings)
|   ├— Application.js      # Application model (Job Applications)
|— routes/
|   ├— userRoutes.js       # Routes for user authentication & management
|   ├— jobRoutes.js        # Routes for job posting & searching
|   ├— applicationRoutes.js  # Routes for applying & managing applications
|— middleware/
|   └— auth.js             # Middleware for authentication (JWT verification)
|— controllers/
|   ├— userController.js     # Handles user registration & login
|   ├— jobController.js      # Handles job posting & searching
|   ├— applicationController.js # Handles job applications
|— .env                    # Environment variables (DB connection, JWT secret)
```

```
|— .gitignore              # Ignore node_modules & .env from Git
|— package.json            # Dependencies & scripts
|— server.js               # Main entry file (Express app setup)
└— README.md               # Project documentation
```

---

**Installation & Setup**

**1. Install Dependencies**

```
npm install express mongoose dotenv bcryptjs jsonwebtoken cors
```

**2. Create .env File**

```
MONGO_URI=your_mongodb_connection_string
JWT_SECRET=your_jwt_secret
```

**3. Start Server**

```
node server.js
```

---

**Code Files**

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

```
const mongoose = require('mongoose');
require('dotenv').config();

const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGO_URI, {
      useNewUrlParser: true,
      useUnifiedTopology: true
    });
    console.log('MongoDB Connected');
```

```
  } catch (error) {
    console.error(error.message);
    process.exit(1);
  }
};

module.exports = connectDB;
```

**models/User.js (User Model)**

```
const mongoose = require('mongoose');
const UserSchema = new mongoose.Schema({
  name: String,
  email: { type: String, unique: true },
  password: String,
  role: { type: String, enum: ['job_seeker', 'employer'], required: true }
});
module.exports = mongoose.model('User', UserSchema);
```

**models/Job.js (Job Model)**

```
const mongoose = require('mongoose');
const JobSchema = new mongoose.Schema({
  title: String,
  description: String,
  requirements: String,
  location: String,
  employer: { type: mongoose.Schema.Types.ObjectId, ref: 'User' }
});
module.exports = mongoose.model('Job', JobSchema);
```

**models/Application.js (Application Model)**

```
const mongoose = require('mongoose');
const ApplicationSchema = new mongoose.Schema({
  job: { type: mongoose.Schema.Types.ObjectId, ref: 'Job' },
  jobSeeker: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
  status: { type: String, enum: ['applied', 'reviewed', 'accepted', 'rejected'], default: 'applied' }
});
module.exports = mongoose.model('Application', ApplicationSchema);
```

**routes/userRoutes.js (User Routes)**

```
const express = require('express');
const { registerUser, loginUser } = require('../controllers/userController');
const router = express.Router();
router.post('/register', registerUser);
router.post('/login', loginUser);
module.exports = router;
```

**controllers/userController.js (User Controller)**

```
const User = require('../models/User');
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');
require('dotenv').config();

exports.registerUser = async (req, res) => {
  const { name, email, password, role } = req.body;
  const hashedPassword = await bcrypt.hash(password, 10);
  const newUser = new User({ name, email, password: hashedPassword, role });
  await newUser.save();
```

```
  res.json({ message: 'User registered successfully' });
};

exports.loginUser = async (req, res) => {
  const { email, password } = req.body;
  const user = await User.findOne({ email });
  if (!user || !(await bcrypt.compare(password, user.password))) {
    return res.status(400).json({ message: 'Invalid credentials' });
  }
  const token = jwt.sign({ id: user._id, role: user.role }, process.env.JWT_SECRET);
  res.json({ token });
};
```

**server.js (Main Entry File)**

```
const express = require('express');
const connectDB = require('./config/db');
const userRoutes = require('./routes/userRoutes');
require('dotenv').config();

const app = express();
connectDB();
app.use(express.json());
app.use('/api/users', userRoutes);

app.listen(5000, () => console.log('Server running on port 5000'));
```

---

**Conclusion**

This project provides a **backend for a Job Recruitment Platform** with:

✅ **User authentication** (bcrypt, JWT)

✅ **Job posting & search**

✅ **Job applications management**

✅ **MongoDB database integration**