7. Creating a Job Board

Creating a Job Board using the MEAN stack (MongoDB, Express.js, Angular, Node.js) is a substantial project. Here's a high-level overview and some code snippets to guide you.

Project Setup and Structure

Set up a new project folder and structure for your Job Board. Install the required Node.js packages and create a basic Angular application.

```
# Create a new Angular application
ng new job-board
```

- ## Backend (Node.js & Express.js)

Create the backend of your Job Board using Node.js and Express.js.

Installation of Packages

Install the necessary packages for Express.js, Mongoose, Passport.js for user authentication, and more.

```
npm install express mongoose passport passport-local bcryptjs jsonwebtoken
```

Setting up Express.js

Create your Express.js server, set up middleware, and handle routes.

- **javascript**

```
// server.js
const express = require('express');
const mongoose = require('mongoose');
const passport = require('passport');
const bodyParser = require('body-parser');
const cors = require('cors');

const app = express();

// Middleware
app.use(bodyParser.json());
app.use(cors());

// Database connection
mongoose.connect('mongodb://localhost/job-board', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
  useCreateIndex: true,
});
```

```
// Passport.js for user authentication
app.use(passport.initialize());
require('./config/passport')(passport);

// Routes
app.use('/api/users', require('./routes/users'));
app.use('/api/jobs', require('./routes/jobs'));

const port = process.env.PORT || 3000;

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

Define Schemas and Models

Create Mongoose schemas and models for users and job listings. These models will define how data is structured and stored in MongoDB.

- **javascript**

```
// models/User.js
const mongoose = require('mongoose');
```

```
const userSchema = new mongoose.Schema({
  username: String,
  password: String,
  // Add more fields as needed
});

module.exports = mongoose.model('User', userSchema);

// Create a similar model for Job
```

Routes

Define routes for user registration, login, job posting, and job search.

- **javascript**

```
// routes/jobs.js
const express = require('express');
const router = express.Router();
const Job = require('../models/Job');

// Create a new job listing
router.post('/post', (req, res) => {
```

```
  // Validate user authentication
  // Save the job listing to the database
  // Return success response
});

// Create a route for job search
router.get('/search', (req, res) => {
  // Implement job search functionality
  // Return job listings based on search criteria
});
```

- ## Frontend (Angular)

Create the frontend of your Job Board using Angular. Design the user interface, implement user registration, job posting, and job search features.

Design and UI

Design the user interface for your Job Board using Angular components, templates, and styles.

User Registration

Create registration forms and user registration components for new users.

Job Posting

Implement components and forms for posting new job listings.

- **typescript**

```
// job-post.component.ts
import { Component } from '@angular/core';
import { JobService } from './job.service';

@Component({
  selector: 'app-job-post',
  templateUrl: './job-post.component.html',
})
export class JobPostComponent {
  jobTitle: string;
  jobDescription: string;

  constructor(private jobService: JobService) {}

  postJob() {
    this.jobService.postJob(this.jobTitle, this.jobDescription);
```

```
  }
}
```

Job Search

Create components for job search functionality, allowing users to search and filter job listings.

MongoDB

Create a MongoDB database to store user profiles and job listings.

Authentication

Set up user authentication using Angular components and Passport.js on the server side. Create registration and login forms.

- **typescript**

```
// auth.service.ts
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root',
})
```

```
export class AuthService {
  constructor(private http: HttpClient) {}

  register(user) {
    return this.http.post('/api/users/register', user);
  }

  login(user) {
    return this.http.post('/api/users/login', user);
  }
}
```

Putting It All Together

Integrate the frontend and backend by making API requests from Angular components to Node.js routes. Ensure that you handle user authentication, job posting, and job search properly.

Building a full-fledged Job Board is a complex project, and you should consider other aspects like job management, job application handling, and scalability as you progress with your project.