Implementing user registration and authentication features using a backend server Node.js

1. Set Up Your Node.js Project:

Make sure you have Node.js and npm (Node Package Manager) installed. Create a new directory for your project and run:

“Npm init”

“Npm install express bcrypt”

1. Create Your Server:

Create a Node.js file (e.g., server.js) and set up your Express server:

Const express = require(‘express’);

Const mongoose = require(‘mongoose’);

Const bcrypt = require(‘bcrypt’);

Const passport = require(‘passport’);

Const LocalStrategy = require(‘passport-local’).Strategy;

Const session = require(‘express-session’);

Const app = express();

Const PORT = process.env.PORT || 3000;

// Connect to MongoDB (you need to have MongoDB installed and running)

Mongoose.connect(‘mongodb://localhost/your-database’, { useNewUrlParser: true, useUnifiedTopology: true });

Mongoose.connection.on(‘error’, console.error);

// Create a User model (in a real project, you’d create a more comprehensive user model)

Const User = mongoose.model(‘User’, {

Username: String,

Password: String,

});

// Passport configuration

Passport.use(new LocalStrategy((username, password, done) => {

User.findOne({ username }, (err, user) => {

If (err) return done(err);

If (!user) return done(null, false, { message: ‘Incorrect username’ });

Bcrypt.compare(password, user.password, (err, res) => {

If (res) return done(null, user);

Return done(null, false, { message: ‘Incorrect password’ });

});

});

}));

Passport.serializeUser((user, done) => {

Done(null, user.id);

});

Passport.deserializeUser((id, done) => {

User.findById(id, (err, user) => {

Done(err, user);

});

});

App.use(express.json());

App.use(session({

Secret: ‘your-secret-key’,

Resave: false,

saveUninitialized: true,

}));

App.use(passport.initialize());

App.use(passport.session());

// User registration

App.post(‘/register’, async (req, res) => {

Try {

Const { username, password } = req.body;

// Check if the username is already in use

Const existingUser = await User.findOne({ username });

If (existingUser) {

Return res.status(400).json({ message: ‘Username already in use’ });

}

// Hash the user’s password

Const hashedPassword = await bcrypt.hash(password, 10);

// Create a new user

Const user = new User({ username, password: hashedPassword });

Await user.save();

Res.status(201).json({ message: ‘User registered successfully’ });

} catch (error) {

Console.error(error);

Res.status(500).json({ message: ‘Error while registering’ });

}

});

// User login

App.post(‘/login’, passport.authenticate(‘local’, {

successRedirect: ‘/dashboard’,

failureRedirect: ‘/login’,

failureFlash: true,

}));

App.get(‘/dashboard’, (req, res) => {

If (req.isAuthenticated()) {

Res.json({ message: ‘You are logged in.’ });

} else {

Res.json({ message: ‘You are not logged in.’ });

}

});

App.listen(PORT, () => {

Console.log(`Server is running on port ${PORT}`);

});

1. Running the Server:

Run your server using:

“Node server.js”