

**Experiment: 10**

**Date: 30/10/2025**

**Unique Id : E0224029**

**Implement advanced features such as user authentication, authorization, and dynamic content updates using JavaScript.**

**Code :**

**Server.js**

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const session = require('express-session');
const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;
const bcrypt = require('bcryptjs');
const cors = require('cors');
const User = require('./models/User'); // Assuming you have a User model
const app = express();
const port = process.env.PORT || 5000;
// Middleware
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
app.use(cors());
app.use(session({
  secret: 'secret',
  resave: true,
```

```

saveUninitialized: true
    ));
app.use(passport.initialize());
app.use(passport.session());
// Passport local strategy
passport.use(new LocalStrategy({ usernameField: 'email' }, (email, password, done) => {
  User.findOne({ email: email }).then(user => {
    if (!user) {
      return done(null, false, { message: 'That email is not registered' });
    }
    bcrypt.compare(password, user.password, (err, isMatch) => {
      if (err) throw err;
      if (isMatch) {
        return done(null, user);
      } else {
        return done(null, false, { message: 'Password incorrect' });
      }
    });
  }).catch(err => console.log(err));
}));
passport.serializeUser((user, done) => {
  done(null, user.id);
});
passport.deserializeUser((id, done) => {
  User.findById(id, (err, user) => {
    done(err, user);
  });
});
// Connect to MongoDB

```

```
mongoose.connect('mongodb://localhost:27017/fullstack-auth', {
  useNewUrlParser: true,
  useUnifiedTopology: true
}).then(() => console.log('MongoDB connected'))
.catch(err => console.log(err));

// Routes

const authRoutes = require('./routes/authRoutes');
app.use('/api/auth', authRoutes);

// Start the server
app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

## **Models/Users.js**

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const userSchema = new Schema({
  username: String,
  email: String,
  password: String
});
module.exports = mongoose.model('User', userSchema);
```

## routes/Authroutes.js

```
const express = require('express');
const router = express.Router();
const passport = require('passport');
const bcrypt = require('bcryptjs');
const User = require('../models/User');

// Register
router.post('/register', (req, res) => {
  const { username, email, password } = req.body;
  User.findOne({ email }).then(user => {
    if (user) {
      return res.status(400).json({ message: 'Email already exists' });
    } else {
      const newUser = new User({ username, email, password });
      bcrypt.genSalt(10, (err, salt) => {
        bcrypt.hash(newUser.password, salt, (err, hash) => {
          if (err) throw err;
          newUser.password = hash;
          newUser.save()
            .then(user => res.json(user))
            .catch(err => console.log(err));
        });
      });
    }
  });
});
```

```
// Login
router.post('/login', (req, res, next) => {
  passport.authenticate('local', (err, user, info) => {
    if (err) throw err;
    if (!user) return res.status(400).json({ message: 'No user exists with that email' });
    req.logIn(user, err => {
      if (err) throw err;
      res.send('Successfully authenticated');
    });
  })(req, res, next);
});

// Logout
router.get('/logout', (req, res) => {
  req.logout();
  res.send('Logged out');
});

module.exports = router;
```

## **Client/src/App.js**

```
import React from 'react';
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';
import Login from './components/Login';
import Register from './components/Register';
import Dashboard from './components/Dashboard';
```

```
function App() {  
  return (  
    <Router>  
      <div className="App">  
        <Switch>  
          <Route exact path="/login" component={Login} />  
          <Route exact path="/register" component={Register} />  
          <Route exact path="/dashboard" component={Dashboard} />  
        </Switch>  
      </div>  
    </Router>  
  );  
}  
  
export default App;
```

## **client/src/components/Login.js**

```
import React, { useState } from 'react';  
import axios from 'axios';  
  
const Login = () => {  
  const [email, setEmail] = useState("");  
  const [password, setPassword] = useState("");
```

```

const handleSubmit = async (e) => {
  e.preventDefault();
  try {
    const response = await axios.post('http://localhost:5000/api/auth/login', {
      email,
      password,
    });
    console.log(response.data);
    // Redirect to dashboard or handle success
  } catch (error) {
    console.error('Login error:', error.response?.data?.message || error.message);
    // Handle error, show message
  }
};

```

```

return (
  <div>
    <h2>Login</h2>
    <form onSubmit={handleSubmit}>
      <input
        type="email"
        placeholder="Email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
        required
      />
      <input
        type="password"
        placeholder="Password"

```

```

        value={password}
        onChange={(e) => setPassword(e.target.value)}
        required
      />
      <button type="submit">Login</button>
    </form>
  </div>
);
};

export default Login;

```

## Create client/src/components/Register.js:

```

import React, { useState } from 'react';
import axios from 'axios';
const Register = () => {
  const [username, setUsername] = useState("");
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [password2, setPassword2] = useState("");
  const handleSubmit = async (e) => {
    e.preventDefault();
    if (password !== password2) {
      console.error('Passwords do not match');
      return;
    }
  }
}

```



```

try {
  const response = await axios.post('http://localhost:5000/api/auth/register', {
    username,
    email,
    password,
  });
  console.log(response.data);
  // Redirect to login or handle success
} catch (error) {
  console.error('Registration error:', error.response?.data?.message || error.message);
  // Handle error, show message
}
};

return (
  <div>
    <h2>Register</h2>
    <form onSubmit={handleSubmit}>
      <input
        type="text"
        placeholder="Username"
        value={username}
        onChange={(e) => setUsername(e.target.value)}
        required
      />
      <input
        type="email"
        placeholder="Email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}

```

```

      required
    />
    <input
      type="password"
      placeholder="Password"
      value={password}
      onChange={(e) => setPassword(e.target.value)}
      required
    />
    <input
      type="password"
      placeholder="Confirm Password"
      value={password2}
      onChange={(e) => setPassword2(e.target.value)}
      required
    />
    <button type="submit">Register</button>
  </form>
</div>
);
};

export default Register;

```

### **client/src/components/Dashboard.js**

```

import React, { useState, useEffect } from 'react';
import axios from 'axios';
const Dashboard = () => {

```

```

const [items, setItems] = useState([]);
useEffect(() => {
  const fetchItems = async () => {
    try {
      const response = await axios.get('http://localhost:5000/api/items');
      setItems(response.data);
    } catch (error) {
      console.error('Error fetching items:', error.message);
    }
  };
  fetchItems();
}, []);
return (
  <div>
    <h2>Dashboard</h2>
    {items.length > 0 ? (
      <ul>
        {items.map((item) => (
          <li key={item._id}>
            {item.name}: {item.description}
          </li>
        ))}
      </ul>
    ) : (
      <p>No items available.</p>
    )}
  </div>
);
};

```

export default Dashboard;

**Output :**

## Login

Email

Password

Login

## Register

Username

Email

Password

Confirm Password

Register

