**NodeReactMySQLStudentPortal**

**Description**

NodeReactMySQLStudentPortal is a secure student portal built with a modern web development stack. It allows students and teachers to register, log in, and manage course participation. This project serves as a **proof-of-concept** for comparing Node.js + React vs. Django + Angular stacks, with a strong emphasis on **cybersecurity** practices such as session management, 2FA, and secure password handling.

**✅ Features**

* 🔐 Secure Login & Registration with bcrypt-hashed passwords
* 🔁 Two-Factor Authentication (2FA) via email
* 🔑 Session-Based Authentication using express-session
* 🔄 Password Reset Flow with secure tokens and expiration
* 🧪 SQL Injection Testing Mode
* 🧾 Role-Based Structure for future Student/Teacher access
* 🌐 HTTPS Enabled using custom SSL certificates
* 📦 Modular architecture with React frontend + Node/Express backend + MySQL

**🛠️ Tech Stack**

* Backend: Node.js, Express.js
* Frontend: React.js
* Database: MySQL (AWS RDS ready)
* Security: bcrypt, express-session
* Email: Nodemailer + Gmail
* Protocol: HTTPS with SSL certs

**⚙️ Installation**

1. Clone the repository:
2. git clone https://github.com/YOUR\_GITHUB\_USERNAME/NodeReactMySQLStudentPortal.git

cd NodeReactMySQLStudentPortal

1. Install Dependencies:
2. cd backend
3. npm install
4. cd ../frontend

npm install

1. Environment Variables: backend/.env
2. DB\_HOST=your\_database\_host
3. DB\_USER=your\_db\_user
4. DB\_PASSWORD=your\_db\_password
5. DB\_NAME=your\_database
6. EMAIL\_USER=your\_gmail@gmail.com
7. EMAIL\_PASS=your\_gmail\_app\_password
8. SESSION\_SECRET=thisisaverysecretkey

REACT\_APP\_API\_URL=https://localhost:3000

frontend/.env

SKIP\_PREFLIGHT\_CHECK=true

HTTPS=true

PORT=3000

SSL\_CRT\_FILE=../backend/ssl/server.cert

SSL\_KEY\_FILE=../backend/ssl/server.key

REACT\_APP\_BACKEND\_URL=https://localhost:3443

1. Start backend:
2. cd backend

node server.js

1. Start frontend:
2. cd frontend

npm start

**🔐 Security Features**

* ✅ 2FA via Email When a user logs in successfully with email & password, a 6-digit verification code is sent to their email.

The code expires after 10 minutes.

The code is verified on /verify-2fa before allowing access.

* 🔁 Password Reset with Tokens Users can request a password reset via /forgot-password.

A secure UUID token is generated and stored with a 1-hour expiry.

A reset link is emailed with a ?token=... parameter.

Upon submission of a new password, the token is validated and deleted to prevent reuse.

* 🧠 Sessions with express-session User sessions are managed using express-session.

Session cookies are:

HttpOnly: inaccessible to JS

Secure: sent only over HTTPS

Expire after 1 hour

Sessions are checked on /api/dashboard-data for route protection.

* 🌐 HTTPS + SSL The app runs both frontend and backend over HTTPS using custom SSL certificates stored in backend/ssl.

You must accept the self-signed certificate in your browser.