📄 Phase 1 – Security Vulnerability Report

Project:	M183 TODO-List Application
Phase:	1
Team:	Tsering L. Anodunkhartsang, Aabish T. Khan
Date:	16.06.2025

🔍 Summary of Identified Vulnerabilities

Old project:

❌ “Passwords were stored as plain text”

❌ “Login used cookies like `res.cookie('username')` which can be modified by the user”

❌ “No CSRF protection at all”

❌ “Anyone could access /admin/users – even if not admin”

❌ “No sorting or search in user list”

#	File	Vulnerability	Severity	Fix implemented
1	Login.js, app.js	Insecure authentication	🔥 high	Replaced insecure cookie-based login with secure express-session using req.session
2	Admin/users.js	Missing access control	🔥 high	Added role-based middleware isadmin() to protect /admin/* routes
3	Forms: ejs templates	Csrf (cross-site request forgery)	🔥 high	Implemented csurf middleware and injected <input type="hidden" name="_csrf" ...> in all forms
4	Admin/users.js	Plaintext password storage	🔥 high	Implemented password hashing with bcrypt on user creation and login
5	Views/dashboard.ejs	Xss (cross-site scripting)	🔥 high	Escaped ejs output (<%= %>) properly — no unescaped (<%- %>) user input
6	All sql queries	Sql injection	⚠️ medium	Used parameterized queries with placeholders (?) Throughout, e.g., conn.query(..., [data])
7	Admin/users.js	Insufficient input validation	⚠️ medium	Currently minimal — needs enhancement (e.g., check for empty username/password, valid role)
8	Dashboard.ejs	Lack of feedback on actions	🟡 low	Ui improvement pending — no success/error toast or redirect messages shown after create/edit/deactivate
9	App.js (optional)	Clickjacking	🟡 low	Not yet fixed — could set x-frame-options: deny header
10	App.js (optional)	Brute force login attempts	🟡 low	Optional improvement — rate limiting can be added to login route using express-rate-limit