Vulnerabilities Found in Partner Project

Regular exp Injection
Constructing a regular expression with unsanitized user input is dangerous as a malicious user may be able to modify the meaning of the expression. In particular, such a user may be able to provide a regular expression fragment that takes exponential time in the worst case, and use that to perform a Denial of Service attack.
FIX:
Database query built from user-controlled sources
If a database query (such as a SQL or NoSQL query) is built from user-provided data without sufficient sanitization, a malicious user may be able to run malicious database queries.
FIX:
In Mongoose, when you use the { field: value } syntax, it automatically performs an equality check, and there's no need to explicitly use $eq
Missing CSRF middleware
Websites that rely on cookie-based authentication may be vulnerable to cross-site request forgery (CSRF). Specifically, a state-changing request should include a secret token so the request can't be forged by an attacker. Otherwise, unwanted requests can be submitted on behalf of a user who visits a malicious website.
Missing rate limiting
HTTP request handlers should not perform expensive operations such as accessing the file system, executing an operating system command or interacting with a database without limiting the rate at which requests are accepted. Otherwise, the application becomes vulnerable to denial-of-service attacks where an attacker can cause the application to crash or become unresponsive by issuing a large number of requests at the same time.

Partner Repo: https://github.com/MihikaNigam/Full-Stack-Web-App-with-Login/tree/database-cookies-https